Universidad Carlos III de Madrid

Universidad Carlos III de Madrid

Trabajo de Fin de Grado

Integration of CROWD SDN-based DMM solution with NI PXI Systems for LTE emulation

Student:
Jason Moorcroft
jason.c.moorcroft@gmail.com

Tutor:
Antonio de la Oliva
aoliva@it.uc3m.es

February 15, 2016

Contents

List of Figures

List of Tables

Abstract

The main purpose of this final degree project was to continue, improve, expand and finish the development of a Software Defined Networking (SDN) based Distributed Mobility Management (DMM) solution demo under the Connectivity management for eneRgy Optimised Wireless Dense networks (CROWD) EU FP7 project[1]. The starting point of the project was an implementation of the CROWD project SDN-based DMM solution, consisting of two separate networks we will call districts, each controlled by an SDN local Controller in the lower level of a two-tier hierarchy, the higher level of which is controlled by a regional Controller. Each district has an OpenFlow controlled configurable backhaul accessed by 802.11 capable Points of Attachment (PoA). The initial stage of the project required extensive debugging and testing of the already existing code, fixing and optimizing the starting functionalities. The demo has since been improved by means of vastly increasing the functionality by implementing:

- **Multi-node support**: provision of mobility support and connectivity to one or more mobile node
- **Mobile node whitelist**: a list of MN's allowed to attach
- **Resource Detection**: the capability of detecting the nodes active in the district
- **Video streaming capability**: provision of mobility support when streaming video
- **Infrastructure on Demand**: resource management, activating or deactivating APs following certain criteria
- **Error control**: sometimes during a handover, certain packets misfire and provide invalid information.

The project has been greatly expanded by integrating:

- **LTE simulation:** A major part of this final project was integrating LTE capabilities with the system.e have three laptops, two representing the UE and one the enB. From the two laptops representing the UE, one runs the GUI used for simulating handovers and the other runs the UE side of the ns-3 program. The eNB runs the eNB side of the ns-3 program, the S-GW and the PDN-GW. If using the ns-3 program for LTE simulation, the LTE connection and the data transmission is completely simulated. If we also use the NI PXI, then LTE connection and data transmission is performed over a real channel using real transceivers.

[1] `http://www.ict-crowd.eu/`

- **Video Server:** The Video Server, implemented in a Raspberry Pi, runs a python UDP server that accepts request messages for video streaming. When a MN wants to send a Video Start request message, a first NOP (No Operation) message must be sent because it will be consumed by the CLC for processing and will never reach the Video Server if there are no rules installed for the MN in the OF Switches. When the Video Start messages reaches the Video Server, it uses a VLC command to start streaming to the address that has sent the request using Real-time Transport Protocol (RTP), a network protocol for delivering audio and video over IP networks. As soon as the request is sent, VLC is started on the MN awaiting the UDP video stream.

- **MaxiNet:** a Dynamic Backhaul Network emulation extension of the Mininet environment, created to span the emulation across several physical machines to allow the emulation of very large software-defined networks. As a part of this final project, we successfully integrated our system with a MaxiNet implementation acting as a blackbox, to validate the reachability of the Video Server through a unknown SDN network.

- **Visualization:** A fundamental part of this project is the integration of real-time visualization of the network and its state. This has been achieved by developing a Web Site with its respective Web Server, that get the network information from the CLCs using an API.

Taking into consideration that this project is a *proof of concept*, in order to prove the correct functionality of the system and measure the performance, fully comprehensive experiments and tests have been carried out throughout the project. The experiments were implemented using bandwidth performance tools, connectivity tools and a network protocol analyzer. Given the experiments carried out and the test results obtained, this project has successfully demonstrated the viability of the solution developed and implemented.

CHAPTER 1

Introduction

1.1 Analysis Of The Problem

Wireless data communication is a constituent part of everyday life for hundreds of millions of people. From social networking to Internet-assisted navigation, from voice and infotainment to online gaming, mobile communication devices have become essential to fully live everyday interpersonal relations as well as participate in nation-wide and world-wide events. The consequence is twofold. First, the number of wireless users is rapidly increasing, the offered load doubling every year, thus yielding a 1000x growth in the next ten years. Second, expecting high-quality services and high data rates is becoming normal rather than exceptional. For instance, considering a density population of 5000 people/km2, which is typical of large European cities like London, Madrid, or Paris, and accounting for 20% of the population being mobile data users, each demanding 1 Mbps, would lead to a demand of 1 Gbps/km2, which can be hardly provided by the current wireless infrastructures. The figure grows further if we consider that the per-user demand is expected to increase ten-fold in the next 5 years. Mobile data traffic increased by 81% in 2013 and by 69% in 2014 and the rate of increase is not slowing down (Figure 1.1 [19]). This has a direct correlation with both the increased accessibility to the internet from mobile devices via WLANs and RANs, and the increase in popularity of applications designed for smartphones and tablets. On top of this, operators are migrating their networks to full IP based networks for both voice and data [11].

The solution to cope with this growing traffic demand necessarily entails using more points of access, by increasing their density (dense network deployments) and/or by using different wireless technologies (heterogeneous deployments). Following this trend, network operators have already started to push for denser deployments, building micro-, pico- and femto-cells, and installing Wi-Fi hotspots in public areas to inject capacity where the data traffic demand is particularly high. These efforts notwithstanding, we argue that increasing the number of points of access alone would not remove capacity and performance bottlenecks. In fact, dense deployments are not necessarily synonymous with higher capacity. The case of smart meters is a key example. It has been recently noticed that the diffusion of meters for gas and electricity, endowed with wireless transmitters using the 2.4 GHz ISM band, is generating erratic behaviour in Wi-Fi home devices in USA.Furthermore, having a large number of deployed access points also influences the energy cost, especially for the network operator. In particular, today's

access points and base stations running at zero-load consume almost as much energy as when running at full capacity. As a result, wireless dense networking can potentially lead to wireless chaos and huge energy waste. The ever-growing demand for wireless connectivity requires more and more capacity, in terms of both data volume and number of connected devices. Wireless, dense, and heterogeneous deployments are today's only viable solution to provide such capacity. Since the design of currently available technologies did not account for density- and heterogeneity-related issues, the latter pose new, unexplored, and promising research questions. Specifically, we claim that using the existing wireless technologies in dense scenarios will require new mechanisms for coordination and cooperation of wireless devices; otherwise we will soon witness the implosion of wireless network performance and the explosion of network operation costs.

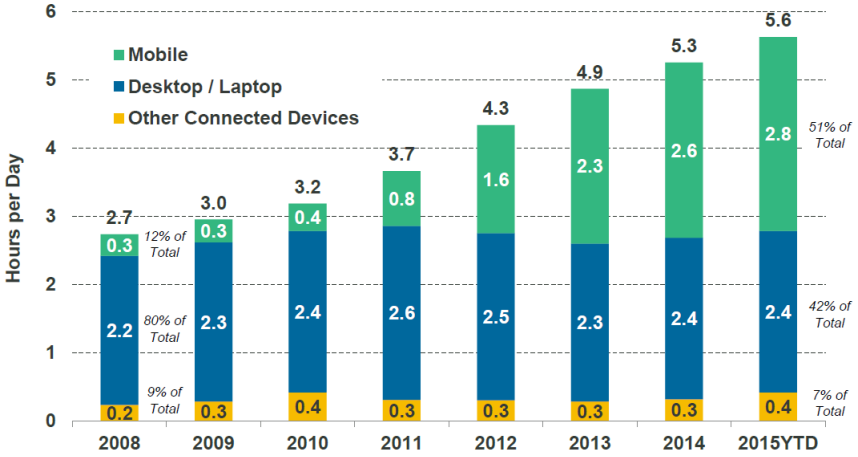


Figure 1.1: Mobile Traffic Data. Adapted from "INTERNET TRENDS 2015 – CODE CONFERENCE"

1.2 The Project

This project encompasses the combination of Software-Defined Networking (SDN) and Distributed Mobility Management (DMM) as a networking solution to the mobility and network management problems in extremely dense wireless networks.

SDN is an approach to networking that involves the separation of the control plane from the data forwarding plane, resulting in the capacity to manage network behaviour dynamically. This is achieved through the abstraction of the forwarding plane using standardized interfaces [1]. The offers the capability of configuring the network backhaul from a centrilized entity without having to configure each element independently. A more detailed description can be found in Chapter 2.1. DMM is a an approach to IP mobility that suggests a distributed network structure compared to a centralized structure to defeat the limitations of existing network protocols that use centrally deployed mobility anchors such as Mobile IP [2]. Some of the limitations DMM strives to solve

are sub-optimal routing, lack of scalability and single point of failure. A broader view of this approach is located in Chapter 2.2.

In section 1.1 we analyse the problems standardized mobility protocols of today will face in ultra dense wireless networks, and the need for optimizing reconfigurability for efficient deployment and management of converged networks. The solution proposed, consisting in a combination of SDN and DMM technologies working together, will be discussed in Chapter 3.

As explained in the Implementation Chapter (Chapter 4),the starting point of this project was an implementation of the CROWD project scenario illustrated in 1.2. In this implementation, there are two districts, both managed by a regional Controller called CROWD Regional Controller, and under the CROWD Regional Controller, each district is locally managed by a CROWD Local Controller and consisting of an OpenFlow controlled configurable backhaul connected to Access Points. We have continued, improved, expanded and finished the development of the implementation by greatly increasing the functionality by implementing (Section 4.3) multi-node mobility support, mobile node whitelist, Infrastructure on Demand, Resource Detection, video streaming mobility, and error control;integrating the solution with a LTE simulating network (Section 4.4.1), a Video Server (Section 4.4.2), a Dynamic Backhaul Network emulated with a software program emulating SDN (Section 4.4.3) and graphical visualization of the network (Section 4.4.4); and lastly by exhaustive testing and debugging of the code (Section 4.5).The implementation of the solution proposed has a number of design alternatives which have been contrasted and compared in Chapter 5.

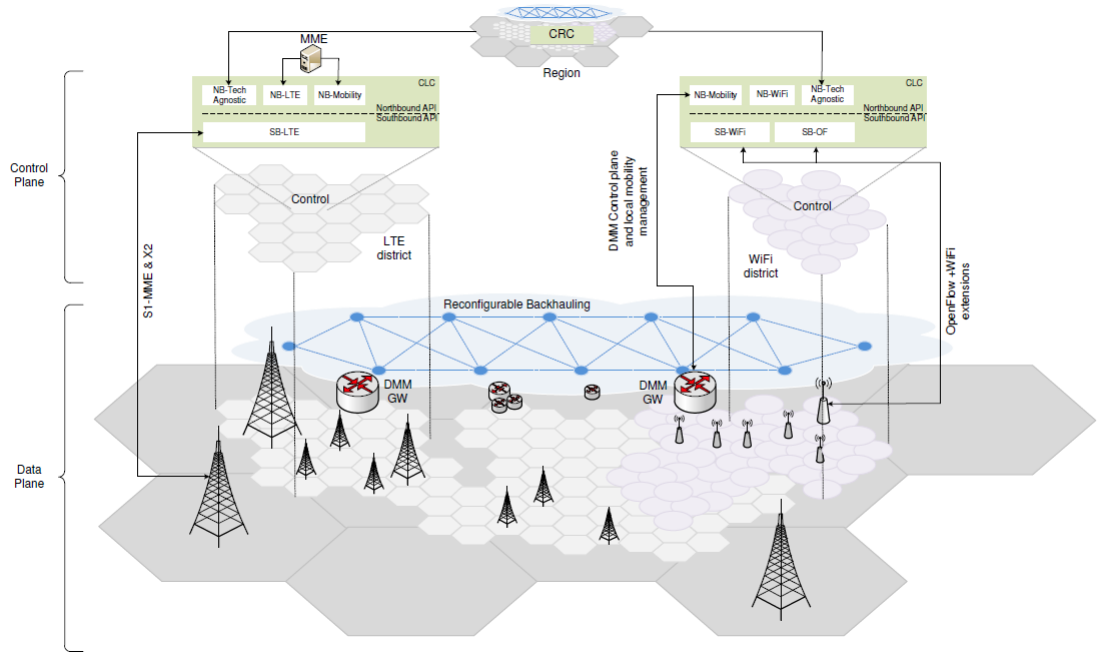


Figure 1.2: CROWD Architecture. Adapted from "CROWD: An SDN Approach for DenseNests"

Exhaustive experiments and tests have been carried out to measure the performance and prove the correct functionality of the system using bandwidth performance tools, connectivity tools and a network protocol analyzer. These tests have been collected in Chapter 6.

A project with such an ambitious scope of objectives as this one requires of a well structured and organized project management. The planning followed can be seen in Chapter 7, represented as a Gantt chart. SDN, a fundamental technology in this project, is under study to explore the unknown area of technological socio-economic-regulatory impact it has and will have, as explained in Chapter 8, and in Chapter 9 we explain that 5G is the future of mobile networking, therefore all mobile communications projects are moving towards that direction. We end this document with the conclusions of this project, found in Chapter 10.

CHAPTER 2

State of the Art

In this Chapter we are going to offer detailed descriptions of the different technologies used in the implementation of this final project.

2.1 Software Defined Networking

Software Defined Networking (SDN) is an approach to networking that revolves around the abstraction between the forwarding and control planes. This facilitates programmatically control over the network in order to operate network services in a deterministic, dynamic and scalable manner [1].

SDN is part of a long history of efforts to make computer networks more programmable, going back more than twenty years. SDN shares similarities with early telephony networks, where there was a clear separation between control and data planes to simplify network management and the deployment of new services [3].

By centralizing the network control, SDN offers flexibility to configure, manage, secure, and optimize network resources via dynamic SDN programs that can be programed independently of the implementation of the network. SDN makes it possible to manage the entire network through intelligent orchestration [4], network administrators can program the behavior of both the traffic and the network in a centralized way, without requiring independently accessing and configuring each of the networks hardware devices [5]. This simplifies network management and the deployment of new applications and protocols, and also makes traffic efficiency and resource management more flexible and configurable.

Figure 2.1 offers a logical representation of the SDN architecture. The Network intelligence resides in the controller, a software-based centralized SDN entity with a global view of the network. APIs between the application layer and the control layer offer the possibility of introducing applications that operate on an abstraction of the network regardless of the details of the implementation.

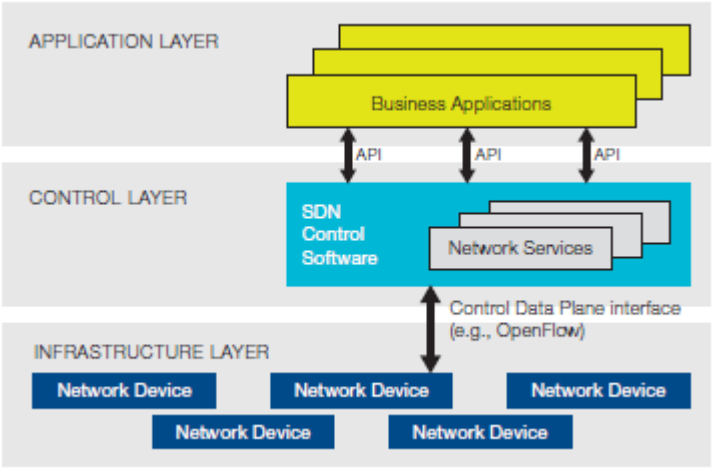


Figure 2.1: Openflow Logical Representation. Adapted from "Software-Defined Networking: The New Norm for Networks"

2.1.1 OpenFlow

The Open Networking Foundation (ONF) is a non-profit organisation that is leading the development of SDN, including the standardisation of the OpenFlow protocol. OpenFlow provides the communication between the control and data planes and is the first standard interface designed specifically for SDN.

OpenFlow is [6]:

- **An architecture:** The OpenFlow architecture consists of a separation of the forwarding plane and the control plane
- **A model:** OpenFlow is a model for match-action at the forwarding plane
- **A protocol:** OpenFlow is a protocol providing the communication

Some of the benefits OpenFlow-based SDN offer to enterprises and carriers are:

- **Centralized management and control of multi-vendor environments:** SDN control software can control any OpenFlow-enabled device.
- **Reduced complexity through automation:** OpenFlow-based SDN offers a flexible network automation and management framework.
- **Higher rate of innovation:** Due to the capacity of network operators to program the network in real time to meet specific needs in a short frame of time.
- **Increased network reliability and security:** Due to the complete visibility and control of the network from a centralized entity.
- **More granular network control:** Capability of setting low level network parameters at a high level.
- **Better end-user experience:** Due to network behaviour adapting to the user's needs, using state information being available to higher-level applications.

Both the SDN controller and the network nodes run the OpenFlow protocol. The OpenFlow protocol's key concept are flows enterprises which identify the network traffic based on rules that are programmed at the SDN control level. These rules establish the way traffic flows through the network through the different network nodes that form it.

OpenFlow Switch

An OpenFlow Switch is a software program or hardware device that consists at least of a Flow Table, an Action associated with each Flow Table Entry and an OpenFlow Secure Channel. The Flow Table performs packet lookups and forwarding, and the OpenFlow Secure Channel connects the OpenFlow switch with the SDN Controller. The switch communicates with the controller and the controller manages the switch via the OpenFlow protocol [7]. Figure 2.2 depicts a logical view of the OpenFlow Switch.

A high percentage of today's routers and Ethernet switches run flow-tables that are usually built from TCAMs, and OpenFlow takes advantage of common functionalities they share. Exploiting this feature, the OpenFlow protocol can program the flowtable regardless of the vendor of the equipment [8].

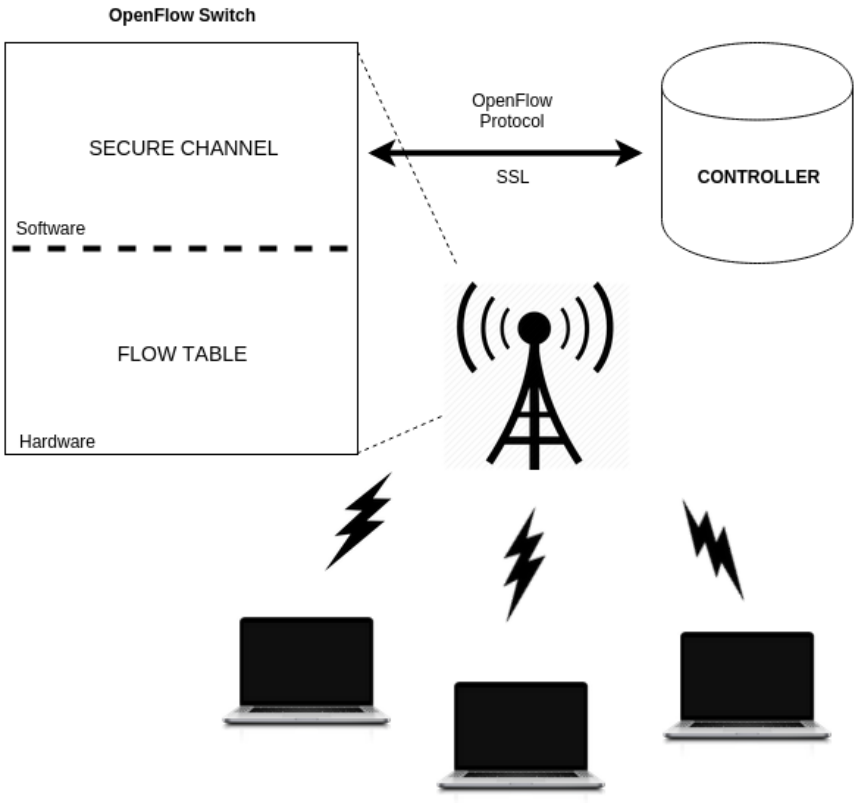


Figure 2.2: OpenFlow Switch Representation

The datapath of an OpenFlow Switch consists of a Flow Table containing three fields:

- **Match information:** Defines the flow.
- **Action:** Defines how the flow should be processed.
- **Statistics:** Keeps track of the number of packets and bytes introduced by the flow and how long since the last packet for control of inactive flows.

The three basic actions (there are 44 in total, but in this project we only use 3) that can be used in the action field of a Flow Table are:

- **Forward:** In order to provide routing capabilities, this action forward packets associated to a given flow to a port or group of ports.
- **Encapsulate:** This encapsulates a packet of a given flow (typically only the first packet) and sends it to the SDN controller for processing. The controller must decide if the flow should be added to the Flow Table.
- **Drop:** This commands to drop packets from a given flow.

2.2 Distributed Mobility Management

Distributed mobility management (DMM) is an approach to IP mobility with the leading objective to evolve from centralized mobility protocols to more distributed protocols, based on already existing centralized mobility standardized protocols such as Mobile IPv6, Hierarchical Mobile IPv6 and Proxy Mobile IPv6. The IETF is leading the standardization of DMM solutions. The DMM working group [1] was formed in March 2012 to address the need for the implementation of a distributed anchoring model in mobile networks. The motivation to study and develop the DMM architectural paradigm resides in the following two challenges [9]:

- Mobile access to the Internet is increasing drastically. The traffic this will generate creates the need for updated requirements for data traffic delivery in mobile core networks. The "flattening" of mobile networks proves to work best for direct communication between closely located peers.
- A study on mobility patterns suggests mobile nodes remain attached to the same point of attachment for relatively long periods of time [10]. IP mobility support is currently designed to always be active, resulting in an unnecessary loss of costs and resources.

In centralized mobility management, the session and forwarding information is kept in a single centralized mobility anchor, which is in charge of forwarding the traffic to the mobile node's current location. The DMM architecture suggests a flat access by means of the distribution of mobility anchors in the data plane so that they are positioned nearer to the user. This optimizes state information management because it avoids unnecessary mechanisms to forward traffic from an old mobility anchor to a new mobility anchor.

Problems that are solved with a DMM implementation are:

[1] `http://datatracker.ietf.org/wg/dmm/`

- **Sub-optimal routes:** Since mobility anchors are fixed at the home address, this is where all traffic will arrive and must be forwarded from to the current mobile node's address. This results in longer end-to-end paths meaning higher delays. With a distributes mobility architecture, as the anchors are located at the very edge of the network, close to the user terminal, data paths tend to be shorter [11].
- **Lack of scalability:** A centralized entity in charge of maintaining mobility context information for each mobile node needs to have enough processing and routing capabilities to deal with all the mobile users' traffic simultaneously. DMM suggests distributing the load among several network entities.
- **Single point of failure and attack:** A centralized entity in charge of maintaining mobility context information for each mobile node means there is a unique point of failure and attack.
- **Unnecessary mobility support:** IP mobility support is usually provided to all mobile nodes, even for users that will not move from the first point of attachment. This also applies to sessions that deal with mobility at an application level or applications that don't need a stable IP address during a handover to maintain session continuity.

The two main approaches to implementing Distributed Mobility Management architectures in modern wireless networks are, on one side, transforming Mobile IPv6 into a distribution-oriented protocol, and, on the other, doing the same for Proxy IPv6. These approaches can be identified as client-based or network-based solutions respectively [12]:

- **Client-based solution:** Based on the Mobile IPv6 protocol, this solution involves deploying multiple home agents at the edge of the network in order to distribute the mobility anchors. This is achieved through the idea that a mobile node can have assigned more than one IP address, meaning that the mobile node can have assigned a new IP address every time it visits an access network.
- **Network-based solution:** Based on the Proxy IPv6 protocol, this solution suggests moving the mobility anchors to the edge of the network, and give them the capability for managing both the control and data plane separately. The ability of managing both planes separately offers the possibility for optimal routing of data traffic.

2.2.1 Mobile IPv6

Mobile IPv6 is a protocol that provides mobility support offering capabilities to mobile nodes so they can move from one Access Point to another without changing the mobile node's home address, the mobile node's permanent address. Without mobility support, once the mobile node leaves the home link, where the mobile node has its subnet prefix assigned (the first point of attachment), packets could not find there way to the mobile node.

Basic Operation

When a MN attaches to a network for the first time, a IPv6 address assigned to the MN called home address (HoA), its permanent unicast routable address, within its home subnet prefix on its home link (HL). The MN can send and receive traffic using conventional Internet routing mechanisms.

When the MN leaves the HL and attaches to a foreign link, it can be addressable using one or more care-of-address (CoA), a unicast routable address associated with a MN while visiting a foreign link, also being able to be obtained using conventional routing mechanisms. While the MN is away from home, it registers its CoA with a router on its home link requesting the router to act as the home agent (HA) for the MN. The HA is in charge of intercepting packets destined to the MN, encapsulates them, and tunnels them to the MN's registered CoA. The request is called a Binding Update (BU), and the home agent replies with a Binding Acknowledgement (BA) message. A bi-directional tunnel between the MN and the HA is created, though which the traffic will flow to and from the Correspondent Node (CN), the MN's communication peer. There exists an optional mechanism called Route Optimization that avoids the sub-optimal routing that involves the the MN sending information to the CN about it's whereabouts (CoA) so the MN and CN can exchange traffic directly independently from the HA.

Figure 2.3 offers an illustration of the basic operation. More detailed information regarding Mobile IPv6 can be found in [2].

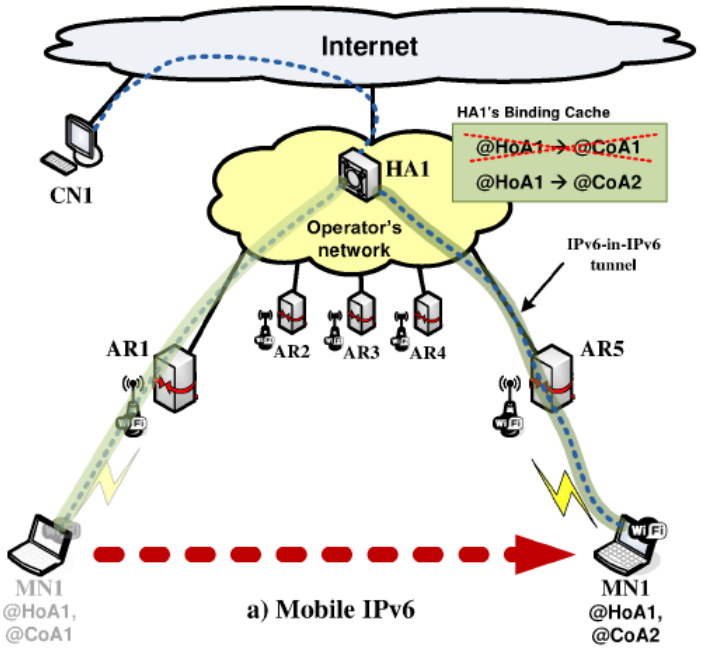


Figure 2.3: Mobile IP. Adapted from "Requirements for Distributed Mobility Management"

2.2.2 Proxy Mobile IPv6

Mobile IPv6 poses the problem of needing the involvement of the client, having to exchange signaling messages with the home agent in order to inform of the mobile node's new location so that the home agent can redirect traffic. The Proxy Mobile IPv6 approach is a Network-based one, excluding the involvement of the host. This is possible thanks to a new entity called proxy mobility agent, that is in charge of the communication with the home agent and manage the mobility on behalf of the client.

Basic Operation

The Proxy Mobile IPv6 introduces the Localized Mobility Anchor (LMA), the equivalent of the home address in Mobile IPv6, and the Mobile Access Gateway (MAG) which is a function on an access router that manages mobility-related signaling on the MN's behalf when attached to its access link. It is responsible for tracking the MN's movements and for signaling the MN's LMA. The LMA has the tasks of keeping track of the MN in it's domain and manage tunnels with every MAG needed for forwarding traffic to the MN.

Once the MN first attaches to an AP in the PMIP domain, the MN sends a RS message which is received by the MAG. The MAG contacts the LMA requesting an IPv6 prefix to assign to the MN with a Proxy Binding Update (PBU) message, and the LMA answers with the prefix in a Proxy Binding Acknowledgement. The MAG sends the MN the prefix via a RA message. The LMA uses the tunnel created (or needs to be created) towards the MAG for traffic to be sent and received by the MN.

When the MN moves to another MAG, the process is similar with the exception that the LMA will identify the MN in its Binding Cache (BC), so the LMA sends the new MAG the same IPv6 prefix that the MN is currently using. The LMA will send traffic destined to the MN through the tunnel towards the new MN.

Figure 2.4 offers an illustration of the basic operation. More detailed information regarding Proxy Mobile IPv6 protocol be found in [13].

2.3 LTE

The Long Term Evolution of 4G UMTS is the result of great technological advances in mobile telecommunications systems,standardized within the 3GPP (3^{rd} Generation Partnership Project) as part of the 3GPP Release 8 feature set. After the development and standardization of GSM (Global System for Mobile communications) belonging to the 2G family, 3G Universal Mobile Telecommunications System (UMTS) came after with the entry of Code Division Multiple Access into the 3GPP evolution track, known as Wideband CDMA (WCDMA). This technology evolved until the arrival of LTE and the adoption of Orthogonal Frequency-Division Multiplexing (OFDM), which is the access technology dominating the latest evolution of all mobile radio standards [14].

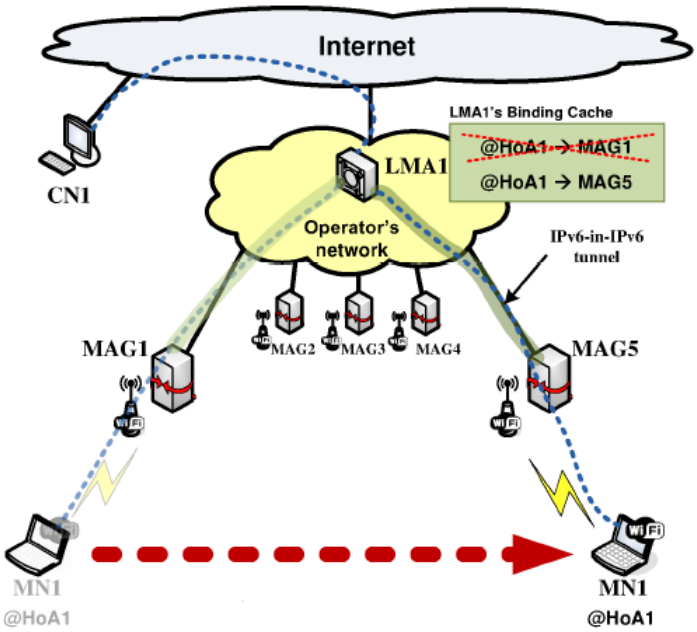


Figure 2.4: Proxy Mobile IP. Adapted from "Requirements for Distributed Mobility Management"

The LTE access network is a network of base stations called evolved NodeB (eNB), generating a flat architecture, and is the access part of the Evolved Packet System (EPS) Figure 2.5.

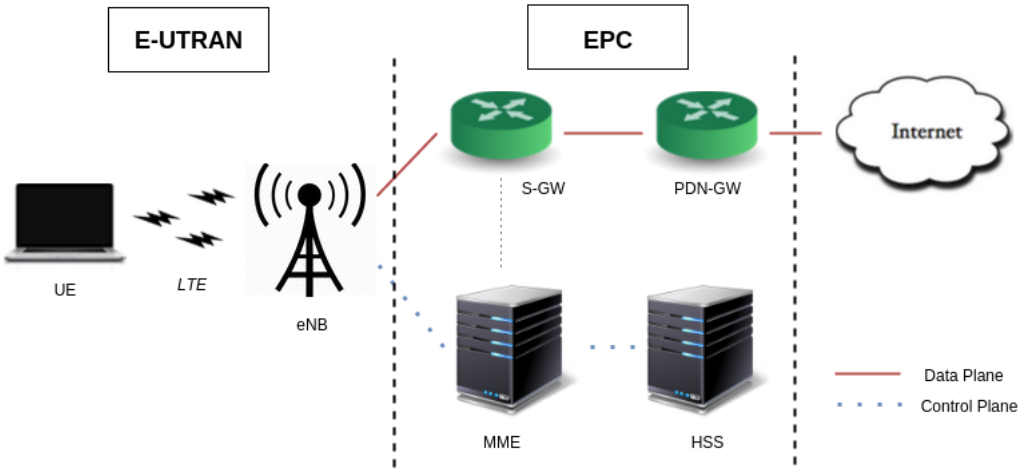


Figure 2.5: Basic EPS Architecture with E-UTRAN access

The latest advancements in LTE technology involve [15]:

- **Small Cell Enhancements**: Higher Order Modulation (256QAM) and Dual connectivity for LTE.
- **Device to Device communication (D2D)**: Offering direct and group communication between terminals.

- **WLAN/3GPP Radio Internetworking**: Integration of WLAN access capabilities into the data transfer.
- **HetNet mobility enhancements**: Enhance the capacity of a cellular network by introducing heterogeneous networks (HetNets) improving overall handover performance and robustness.
- **Smart Congestion Mitigation (SCM)**: For congestion control.

2.3.1 LTE emulation

LTE emulation can be performed with a combination of ns-3, a discrete event network simulator used to create an open simulation environment for computer networking research and NI PXI systems [16] (Figure 2.6).

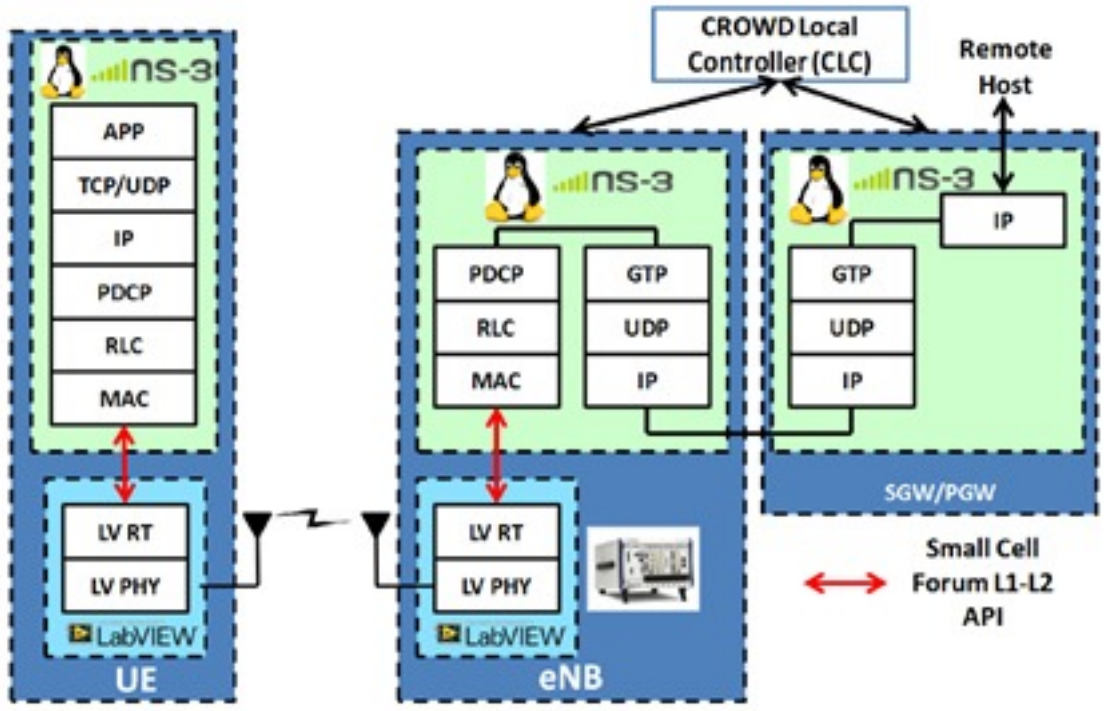


Figure 2.6: LTE emulation with ns-3 and NI PXI. Adapted from "LabVIEW Based Platform for Prototyping Dense LTE Networks in CROWD Project"

NI PXI

PXI is a rugged PC-based platform for measurement and automation systems. PXI combines PCI electrical-bus features with the modular, Eurocard packaging of CompactPCI and then adds specialized synchronization buses and key software features. PXI is both a high-performance and low-cost deployment platform for applications such as manufacturing test, military and aerospace, machine monitoring, automotive, and industrial test. Developed in 1997 and launched in 1998, PXI is an open industry

standard governed by the PXI Systems Alliance (PXISA), a group of more than 70 companies chartered to promote the PXI standard, ensure interoperability, and maintain the PXI specification [17].

ns-3

ns-3 is a discrete-event network simulator, targeted primarily for research and educational use. ns-3 is free software, licensed under the GNU GPLv2 license, and is publicly available for research, development, and use. The goal of the ns-3 project is to develop a preferred, open simulation environment for networking research: it should be aligned with the simulation needs of modern networking research and should encourage community contribution, peer review, and validation of the software [18].

CHAPTER 3

Solution Proposed

3.1 CROWD Architecture

The solution proposed to deal with the problems discussed in Chapter 1.1, is a Connectivity management for eneRgy Optimised Wireless Dense networks (CROWD) initiative, that uses a combination of Software Defined Networking and Distributed Mobility Management techniques.

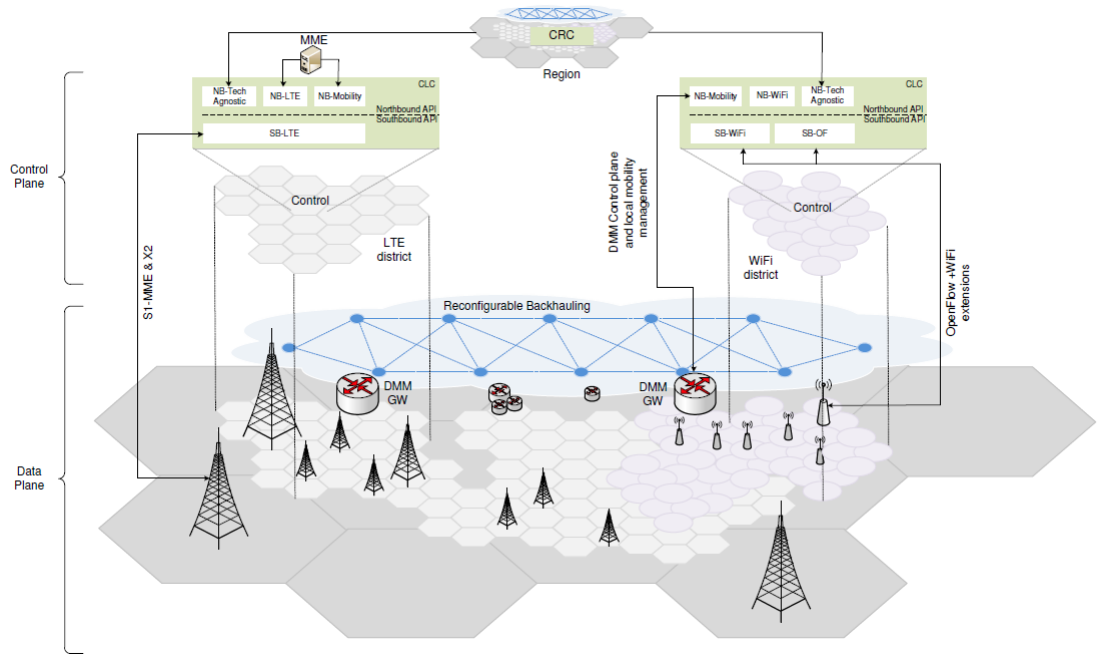


Figure 3.1: CROWD Architecture. Adapted from "CROWD: An SDN Approach for DenseNests"

The CROWD Architecture (Figure 3.1) is structured into two logical tiers, each managed by a CROWD Controller: Districts with a limited, but fine grain scope for short time scales and operated by a CROWD Local Controller (CLC), and Regions

with a broader but more coarse grain scope for long time scales and operated by a CROWD Regional Controller (CRC).

Figure 3.2 shows the network control layer of the CROWD architecture. A district consists of:

- **Base stations:** WiFi Access Points in the case of a WLAN District or LTE eNBs in the case of a RAN District.
- **CROWD Local Controller:** An SDN controller that can take fast, short time decisions on a limited but fine grain scope.
- **Backhaul:** A reconfigurable interconnected backhaul using an open protocol such as OpenFlow for control management
- **DMM Gateways:** They are the default gateway and Mobility Anchor for a Mobile Node, that have Distributed Mobility Management capabilities and connect the district to other districts and networks

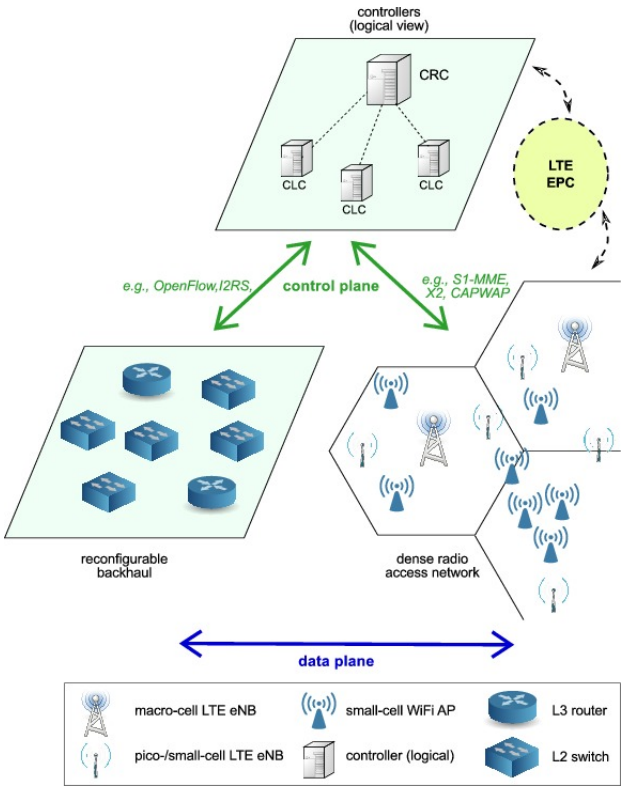


Figure 3.2: CROWD Architecture components. Adapted from "An SDN-based Network Architecture for Extremely Dense Wireless Networks"

The context chosen for the solution is video streaming, having taken into consideration that it is a very highly demanded application in today's wireless networks.

3.2 Network-Based Mobility Solution

The Network-Based Mobility solution is activated when a Mobile Node (MN) associates to a point of attachment in a CROWD District. This solution, that uses SDN mechanisms to reconfigure the backhaul at a layer 2 level, is transparent to the MN, that perceives the path to the DMM Gateway (DMMGW) as a single hop at a layer3 (Figure 3.3).

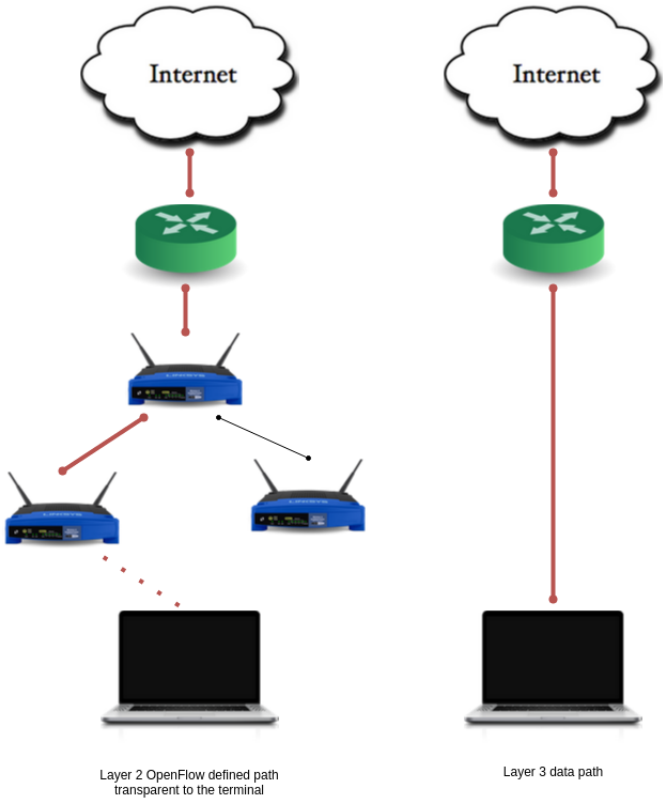


Figure 3.3: The SDN network is transparent to the terminal

In the following sections we will discuss key scenarios contemplated by CROWD and their respective protocols, starting from the initial first point of attachment.

3.2.1 Attachment

When the MN associates to an Access Point (AP) belonging to a CROWD District, the AP generates a Logical Link Control (LLC) frame that is encapsulated in an OpenFlow message and sent to the CLC. The CLC will identify the type of message and process it by checking if the MN is contained in the local Binding Cache (BC). Because this is the first time the MN has associated to a AP in this district, the local BC does not contain a Binding Cache Entry (BCE) for the MN. The CLC will then communicate with the CRC to check if the MN is contained in the regional BC. Being the MN's first interaction with the CROWD network, the CRC's BC has no BCE regarding the MN,

so the CRC will reply with this information. The CLC now knows that the MN is new.

The CLC will extract a prefix from IPv6 prefix pool and assign it to the MN. A DMM Gateway (DMMGW) is also assigned to the MN; the CLC will install the static routing for the prefix in the assigned DMMGW. The CLC saves the MN's prefix, assigned DMMGW and terminal identification (e.g., MAC address) as a BCE in the local BCE, and sends this information to the CRC so it can also include a BCE for the MN.

The CLC generates a Router Advertisement (RA) that contains the assigned prefix and DMMGW and sends it to the MN on behalf of the DMMGW, rendering the attachment process complete.

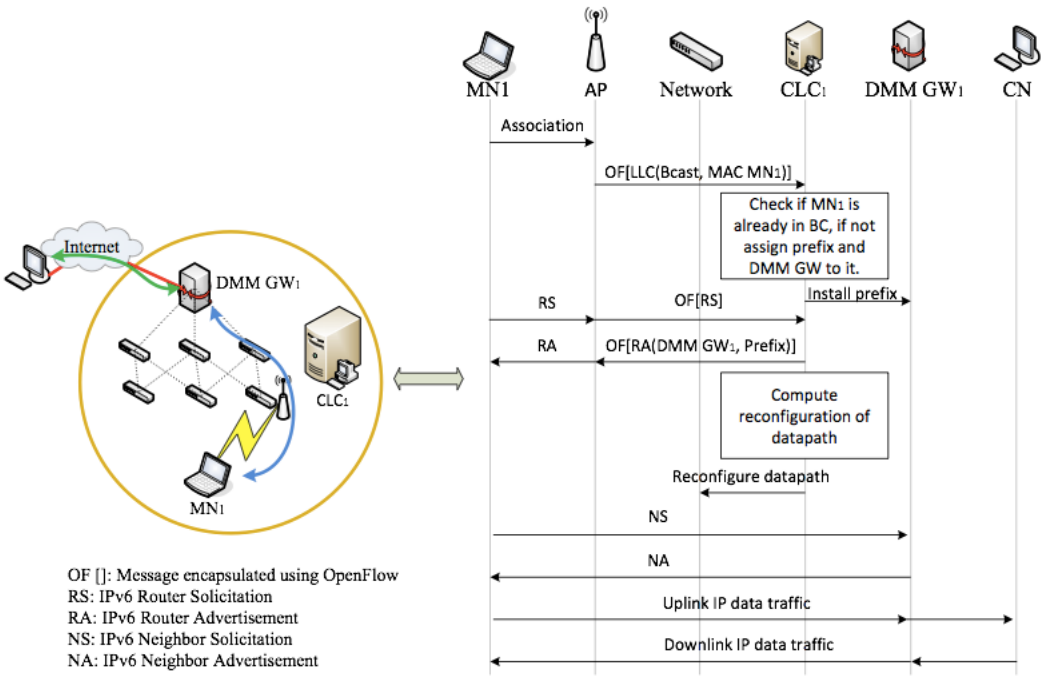


Figure 3.4: CROWD District attachment. Adapted from "An SDN-based Network Architecture for Extremely Dense Wireless Networks,"

3.2.2 Video streaming

To start the video streaming once the attachment process has concluded, the MN sends a Start Video request to the CN acting as a Video Server. The request initially reaches the AP: when a OpenFlow controlled backhaul node, such as an AP or an OpenFlow Switch, does not have a rule associated with the traffic that the MN is generating, is is forwarded to the CLC for processing. As the AP does not have any rules created for the packet sent by the MN, the AP forwards the request to the CLC.

The CLC identifies the packet as an IPv6 packet with source at the MN. The CLC first checks the BC for an entry containing the information of the MN, with two possible

outcomes: if the BC has no entry for the MN, the packet is sent back to the network and the processing terminates, and if the BC does in fact have an entry for the MN, the processing continues.

After spotting the MN in the BC, the CLC computes the upstream data path that the packets need to follow between the MN and the DMMGW, generating the OpenFlow rules to send to the backhaul nodes participating in the data path. Once the nodes are reconfigured with these new rules, traffic can flow upstream from the MN via the DMMGW.

The Start Video request sent by the MN to start the video streaming is formed by 2 IPv6 packets, the first being a No Operation (NOP) packet and the second containing the actual Start Video message as well as the properties defined by the user. This is because the first packet that is sent by the MN is consumed by the CLC for processing. By the time the second packet is sent, the rules have been set in the corresponding nodes, and the subsequent Neighbor Solicitations (NS) and Neighbor Advertisements (NA) have been exchanged, so it can successfully travel towards the Video Server.

Once the Video Server receives the message, the video streaming towards the MN is started. The first packet of the stream reaches the DMMGW, that will send a Neighbor Solicitation for the MN, reaching the first OpenFlow node in the district. Following what has been discussed before, the rules for the downstream traffic forwarded by the DMMGW have not been yet created, so the Neighbor Solicitation is forwarded to the CLC for processing. The CLC identifies the packet as an IPv6 packet with source at the Video Server and the destination at the MN. Similar to the upstream case, the CLC first checks the BC for an entry containing the information of the MN.

The CLC computes the downstream data path that the packets need to follow between the DMMGW and the MN, generating the OpenFlow rules to send to the backhaul nodes of interest. Once the nodes are reconfigured with these new rules, traffic can flow downstream from the Video Server to the MN via the DMMGW.

3.2.3 Intradistrict mobility

When the MN moves to an AP in the same CROWD District, it is considered a Intradistrict handover. Once associated, the AP generates a Logical Link Control (LLC) frame that is encapsulated in an OpenFlow message and sent to the CLC. The CLC will identify the type of message and process it by checking if the MN is contained in the local Binding Cache (BC). Because the MN performed an Intradistrict handover, the MN has previously been associated to a AP in this district, therefore the local BC contains a Binding Cache Entry (BCE) for the MN. The CLC updates its BC with the AP the MN is currently connected to, computes the old data path, and commands the nodes in the old data path to delete the rules related to the MN. At this point there should be no rules established for downstream or upstream traffic to or from the MN.

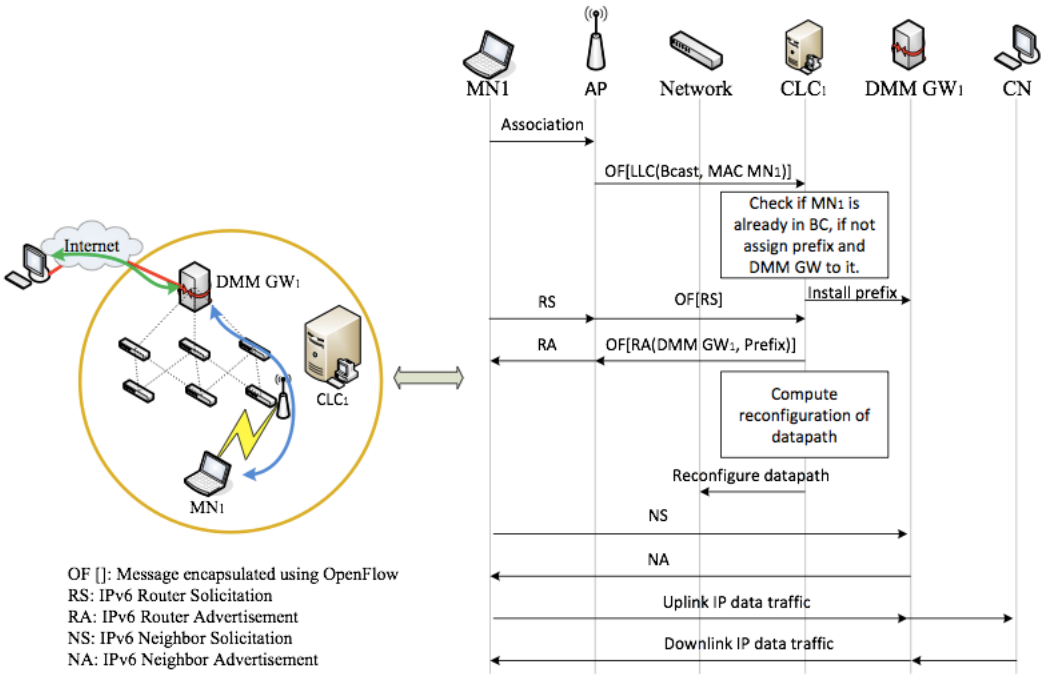


Figure 3.5: Connection provision between the MN and the Video Server. Adapted from "An SDN-based Network Architecture for Extremely Dense Wireless Networks"

The Video Server is still streaming, generating traffic towards the MN. As soon as the rules have been deleted in the old data path, the packet that arrives at the DMMGW from the Video Server is forwarded to the MN and reaches the first OpenFlow node. Following what has been discussed before, the rules for the downstream traffic forwarded by the DMMGW have not been yet created, so the packet is forwarded to the CLC for processing. The CLC identifies the packet as an IPv6 packet with source at the Video Server and the destination at the MN. The CLC checks the BC for an entry containing the information of the MN.

The CLC computes the downstream data path that the packets need to follow between the DMMGW and the MN, generating the OpenFlow rules to send to the backhaul nodes of interest. Once the nodes are reconfigured with these new rules, traffic can flow downstream from the Video Server to the MN via the DMMGW allowing the user to continue to view the content in an non-interrupted fashion.

3.2.4 Interdistrict mobility

An Interdistrict handover occurs when a MN moves to an AP in a new CROWD District when it was previously connected to an AP in another CROWD District. When the MN associates to an Access Point (AP) belonging to a CROWD District, the AP generates a Logical Link Control (LLC) frame that is encapsulated in an OpenFlow message and sent to the CLC. The CLC will identify the type of message and process it by checking if the MN is contained in the local Binding Cache (BC). Because this

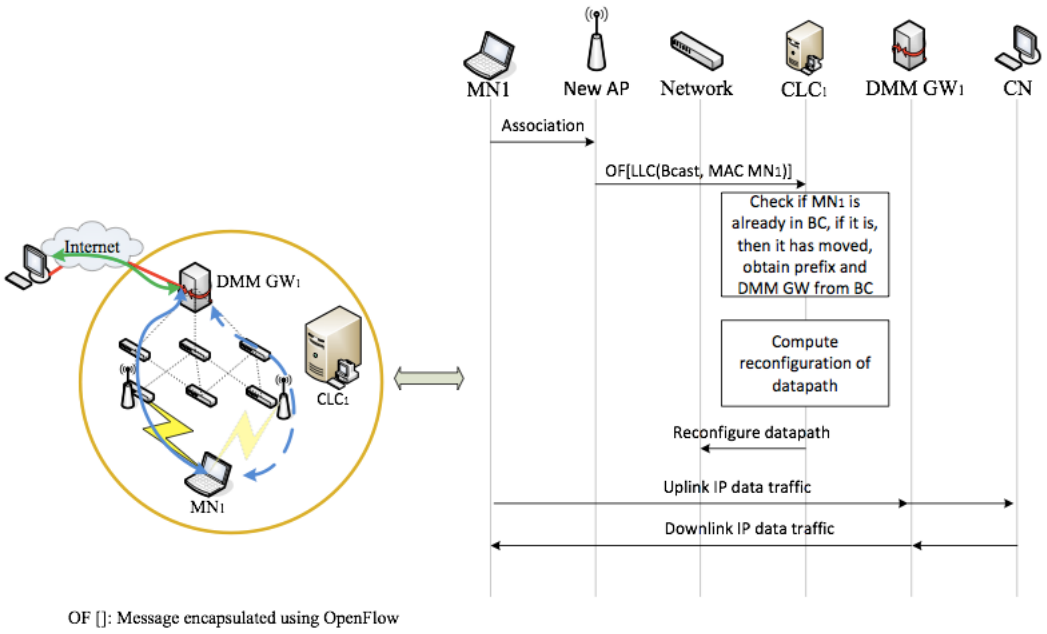


Figure 3.6: CROWD Intradistrict handover. Adapted from "An SDN-based Network Architecture for Extremely Dense Wireless Networks"

is the first time the MN has associated to a AP in this district, the local BC does not contain a Binding Cache Entry (BCE) for the MN. The CLC will then communicate with the CRC to check if the MN is contained in the regional BC. As the MN used to be connected to an AP in another CROWD District, the CRC has a BCE containing this information. Two operations are performed in parallel:

- The CRC informs the old CLC (CLC1) that the MN has performed a Interdistrict handover. The CLC1 will command the DMMGW associated to the MN (DMMGW1) to create a tunnel towards the DMMGW of the new district (DMMGW2) to forward traffic directed to the MN; The DMMGW1 becomes the Mobility Anchor for the MN
- The CRC informs the new CLC (CLC2) that the MN has performed a Interdistrict handover. The CLC2 will command the DMMGW2 to create a tunnel towards the DMMGW to receive traffic directed to the MN

The Video Server is still streaming, generating traffic towards the MN. As soon as the tunnels between the DMMGW1 and DMMGW2 are set, traffic will start to arrive at the DMMGW2 via the tunnel. The first packet that arrives at the DMMGW from the Video Server is forwarded to the MN and reaches the first OpenFlow node. Following what has been discussed before, the rules for the downstream traffic forwarded by the DMMGW have not been yet created, so the packet is forwarded to the CLC for processing. The CLC identifies the packet as an IPv6 packet with source at the Video Server and the destination at the MN. The CLC checks the BC for an entry containing the information of the MN.

The CLC computes the downstream data path that the packets need to follow between the DMMGW and the MN, generating the OpenFlow rules to send to the backhaul nodes of interest. Once the nodes are reconfigured with these new rules, traffic can flow downstream from the Video Server to the MN via the DMMGW allowing the user to continue to view the content in an non-interrupted fashion.

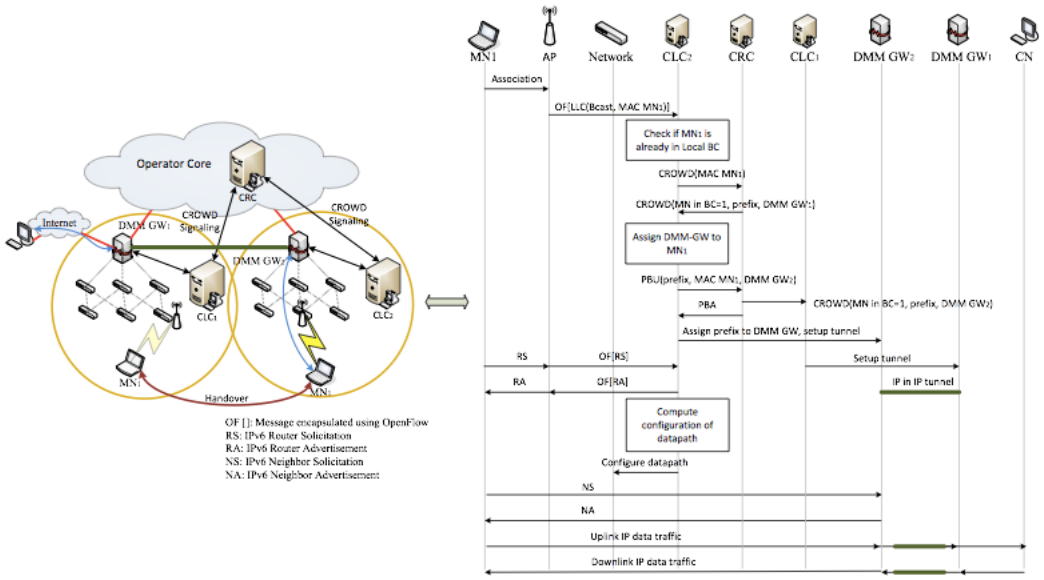


Figure 3.7: CROWD Interdistrict handover. Adapted from "An SDN-based Network Architecture for Extremely Dense Wireless Networks"

3.3 Mobility to a non-CROWD network

To provide mobility towards a non-CROWD network without breaking the connection session, the Client-Based Mobility DMM solution is used. The MN, considered a UE (User Equipment) in LTE terminology, associates to a eNB Base Station.

Once the association process has finished, the terminal requests a prefix of a DHCP lease, obtaining either an IPv4 or an IPv6 address, after which the MN communicates with the CRC, acting as Home Address (HA) for the MN, indicating its location with a Binding Update (BU) message containing the newly assigned IP address as Care-of-Address (CoA). After this, the CRC checks its BC for the CLCs managing the mobility for the MN by checking the prefixes it uses as HA. The CRC sends these CLCs a message so that they can subsequently request the MN's associated DMMGWs to create the corresponding IPv6-in-IPv6 or IPv6-in-IPv4 tunnels. This enables the connectivity continuity for the video-streaming CN towards the MN.

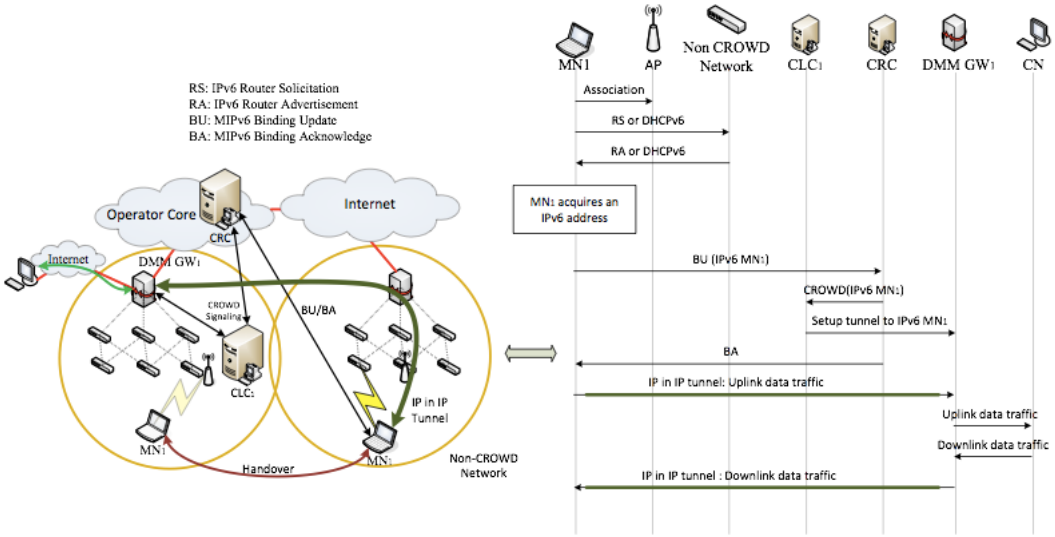


Figure 3.8: Handover to a non-CROWD LTE network. Adapted from "An SDN-based Network Architecture for Extremely Dense Wireless Networks"

3.4 Visualization of the global network

For network administrators, it is of vital importance to have access to a global real-time view of the network. The solution proposed is a graphical interface with a real-time visualization of the network, showing the current network topology and other information at a quick glance such as:

- **MNs points of attachment**
- **Data paths installed**
- **Handovers**
- **Resources on Demand**
- **Tunnels created and deleted**

3.5 Resources on Demand

Given a low density of MNs connected at to a district, it can be inefficient and a waste of resources to have all APs switched on. Another case of inefficiency of resources is having a high density of MNs connected to an AP of a district and a low density of MNs connected to an AP of the same district. The solution proposed to deal with this is to create an API for the CROWD Local Controller for resource and energy efficiency management. Two scenarios are considered:

- **Enable OpenFlow nodes**: When there is a high density of MNs attached to the district, and there are APs powered off and not being used. Once switched on, force handovers on a number of MNs to the new APs

- **Disable OpenFlow nodes**: When there are too many APs switched on due too a low density of MNs attached to the district, force MNs to handover to dedicated APs and switch off the ones left redundant

3.6 High-level business process

The approach proposed in this Chapter enables the creation of a market of control applications developed by third-party services providers [21] as illustrated in Figure 3.9. These control applications can be sold to telecom operators soy they can control the network at their leisure. Using the specific example detailed in the Figure, an operator can used a pre-defined set of policies to enforce a customized energy efficiency management mechanism. The policies and the control application are passed through a policy translator that parses the policies to a low level language so the control application can be configured. The dispatcher is then in charge of installing the configured control application on all the CRCs of the telecom operator's network.

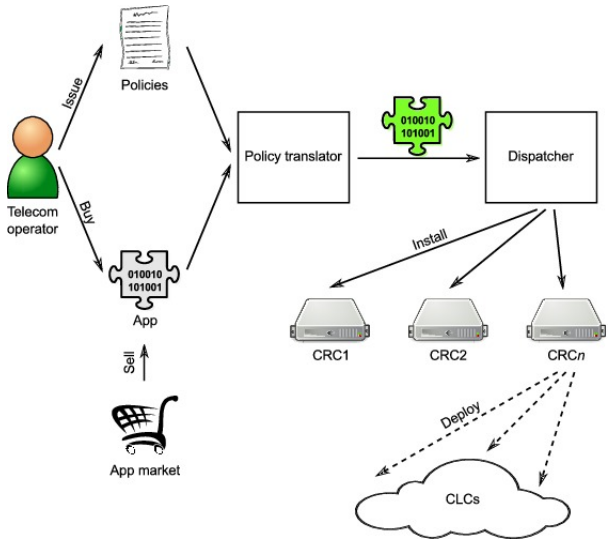


Figure 3.9: CROWD Architecture components. Adapted from "An SDN-based Network Architecture for Extremely Dense Wireless Networks"

CHAPTER 4

Implementation

In Chapter 3 we described the solution proposed in this project to deal with the problems described in Chapter 1.1. In this Chapter, we describe our implementation of the solution proposed in which we illustrate the architecture used, the basic functionality of the system, and the integration plus improvements performed to add value to the original state of the project.

4.1 Testbed Architecture

Photos of the testbed can be found in Figures 4.1, 4.2 and 4.3. In Figure 4.4 we show the Network diagram of the testbed.

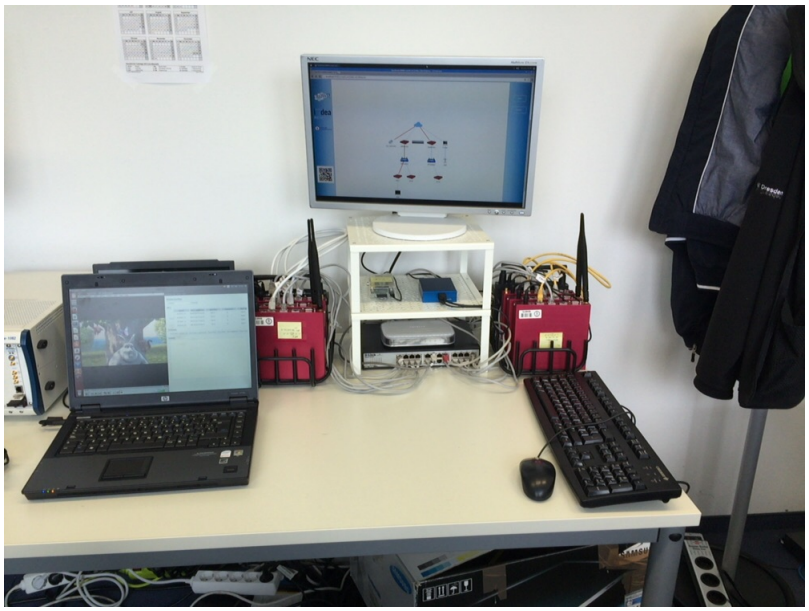


Figure 4.1: Photo of the Testbed

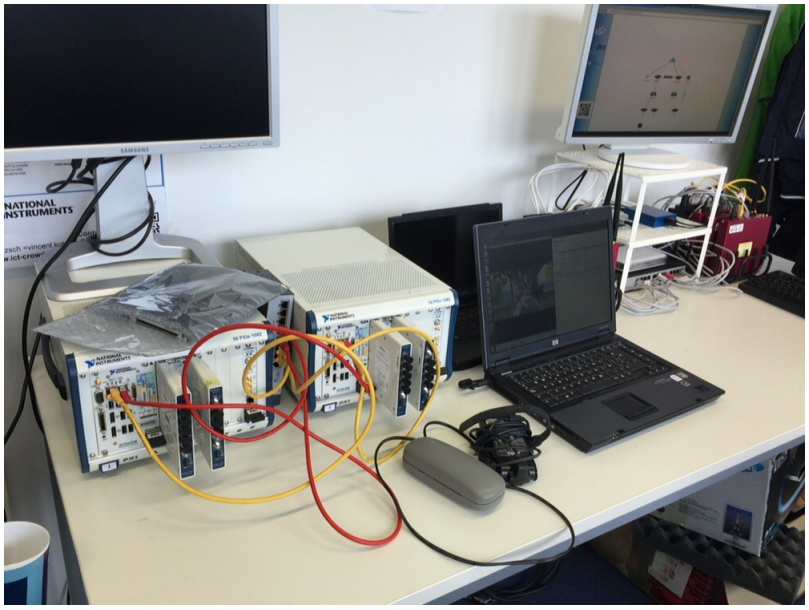


Figure 4.2: Photo of Testbed connected to the NI PXI system

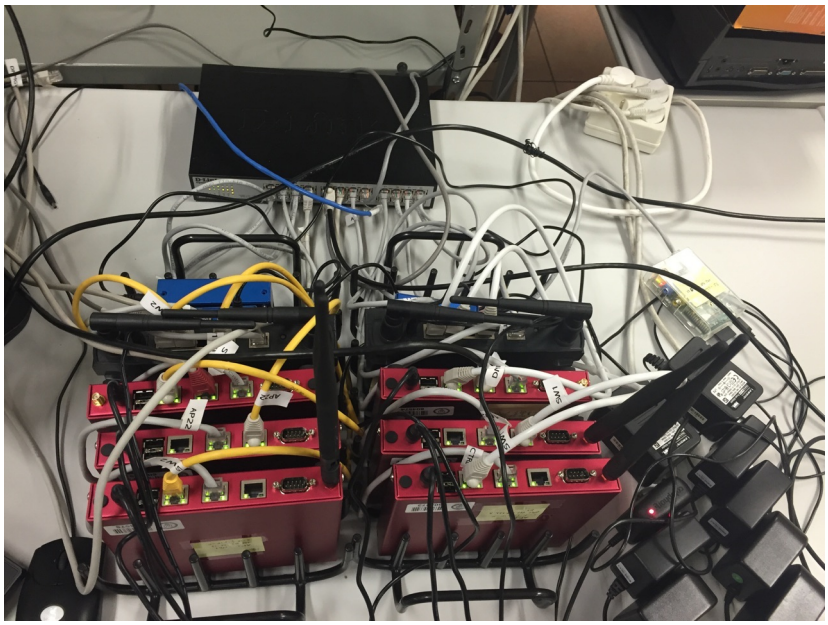


Figure 4.3: Cable layout

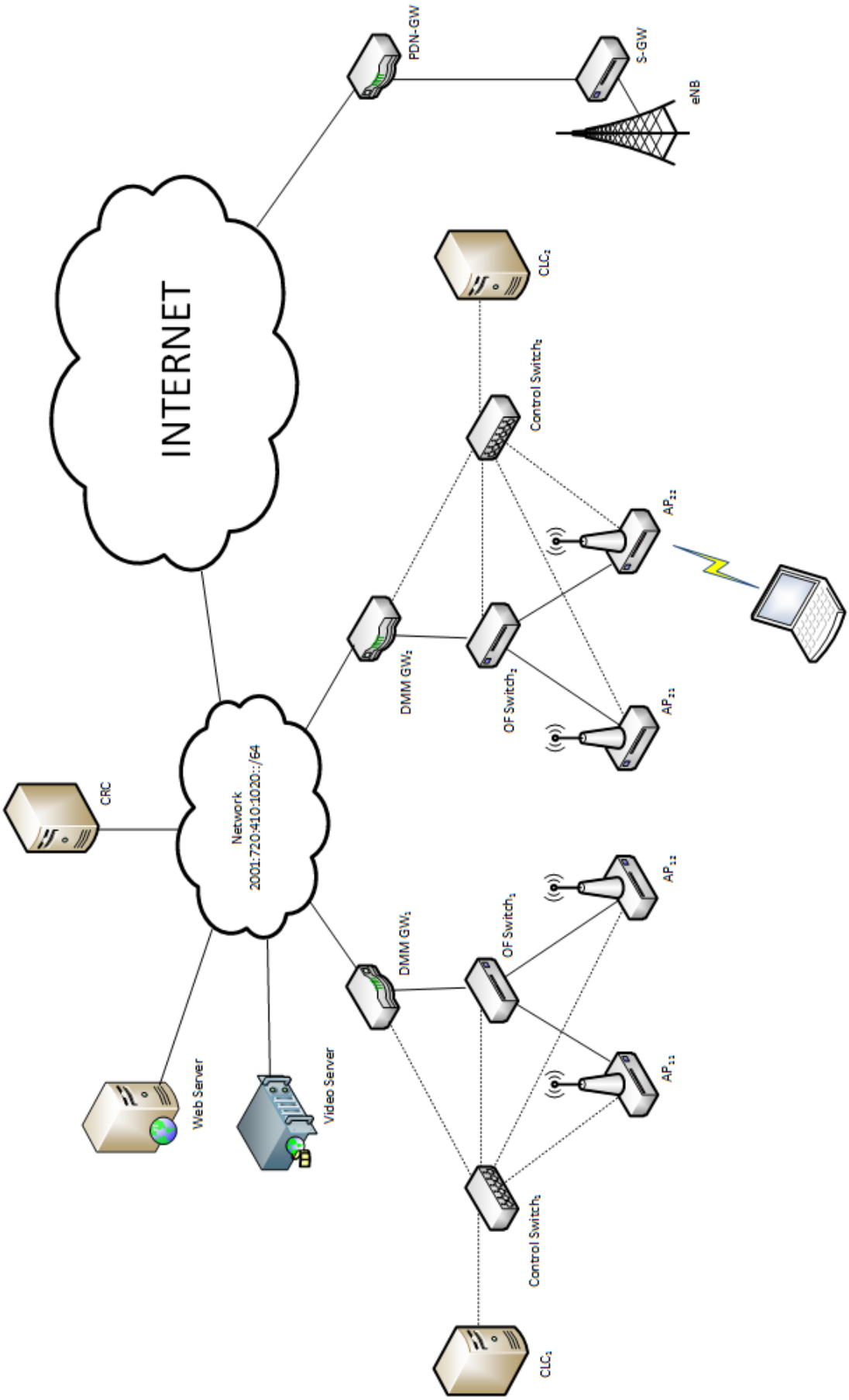


Figure 4.4: Testbed diagram

4.1.1 Hardware

- **CROWD Regional Controller (CRC)**: The CRC is implemented using a MinnowBoard, a compact and affordable open source hardware reference platform that puts the power of a 64-bit Intel® Atom™ E38xx Series System on a Chip (SoC) in a small versatile form-factor. The MinnowBoard.org Foundation is a is a US-based non-profit providing education and promotion of the design and use of open-source software and hardware in embedded computing, on Intel Architecture.
- **CROWD Local Controller (CLC)**:The CLCs are also implemented with MinnowBoards
- **Mobile Node + UE + eNB:** The MNs, UE and eNB are implemented using Ubuntu capable laptops
- **DMM Gateway:** The DMMGW are implemented using Alix PCs
- **Access Point:** The APs are implemented using Alix PCs
- **OpenFlow Switch:** The OF Switches are implemented using Linksys WRT54GL
- **Video Server:** The Video Server is implemented using a Raspberry Pi
- **LTE transceivers:** The LTE transceivers are implemented using NI PXI equipment
- **APC Powerstrip:** Used for the Infrastructure on Demand implementation

4.1.2 Software

- **Ryu [23]**: Ryu Controller is an open, software-defined networking (SDN) Controller designed to increase the agility of the network by making it easy to manage and adapt how traffic is handled. The Ryu Controller provides software components, with well-defined application program interfaces (APIs), that make it easy for developers to create new network management and control applications.
- **Open vSwitch [24]**: Open vSwitch is a production quality, multi-layer virtual switch licensed under the open source Apache 2.0 license. It is designed to enable massive network automation through programmatic extension, while still supporting standard management interfaces and protocols
- **VLC [25]**: VLC is a free and open source cross-platform multimedia player and framework that plays most multimedia files as well as DVDs, Audio CDs, VCDs, and various streaming protocols. VLC can be used as a server and as a client to stream and receive network streams. VLC is able to stream all that it can read.
- **ns-3**: See Chapter 2.3.1
- **Pantou [26]**: Pantou turns a commercial wireless router/Access Point to an OpenFlow-enabled switch. OpenFlow is implemented as an application on top of OpenWrt. Pantou is based on the BackFire OpenWrt release (Linux 2.6.32). The OpenFlow module is based on the Stanford reference implementation (userspace).

- **Scapy** [27]: Scapy is a powerful interactive packet manipulation program. It is able to forge or decode packets of a wide number of protocols, send them on the wire, capture them, match requests and replies, and much more. It can easily handle most classical tasks like scanning, tracerouting, probing, unit tests, attacks or network discovery.

4.2 Basic Functionality

In this section we describe the basic functionality of the system, from the moment the MN joins a CROWD District and exchanges traffic with a Correspondent Node (CN) for the first time, and describing the processes performed when the MN moves to an AP in the same district, to another CROWD District or a non-CROWD district.

4.2.1 Attachment

Here we will describe the implementation of the process executed when a MN attaches to a CROWD District for the first time (represented in Figure 4.8).

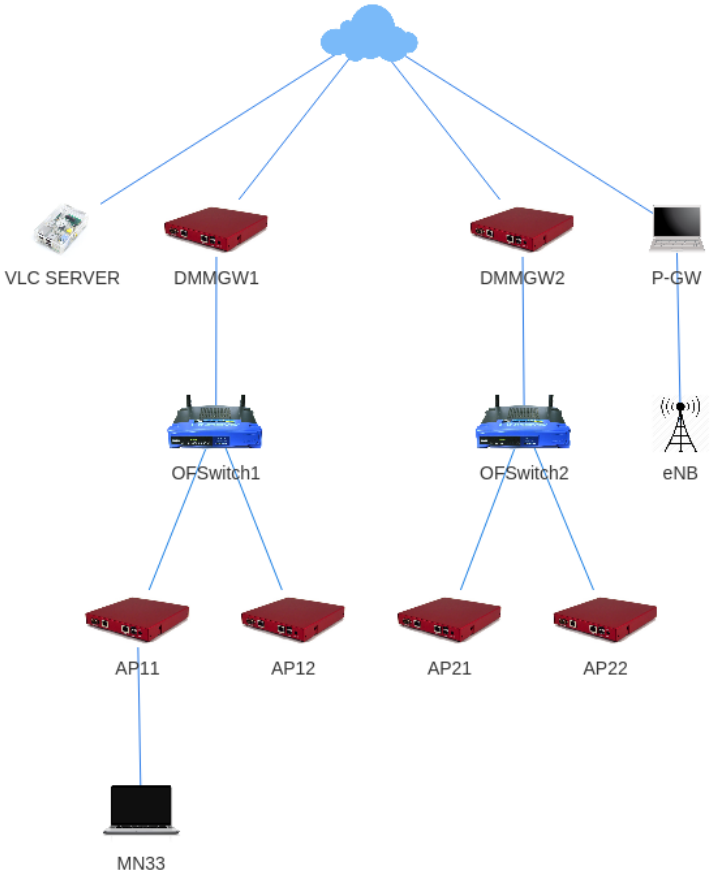


Figure 4.5: MN Attachment visualization

When an MN associates with an AP, the AP generates a LLC message which is forwarded by the OF Switch module running in the AP to the CLC of the same district via the Ryu framework encapsulated in an OpenFlow message, due to the OpenFlow Table not containing an entry instructing the OpenFlow Switch what to do with LLC frames. An OpenFlow Packet-in Event (listing 4.1) will be raised in the CLC once it has received the message, executing the CLC Packet-in application following these steps:

Listing 4.1: PacketIn OpenFlow Event Handler

```
@set_ev_cls(ofp_event.EventOFPPacketIn, MAIN_DISPATCHER)
   def _packet_in_handler(self, ev):
```

Listing 4.2: Data extraction from the OpenFlow message

```
@set_ev_cls(ofp_event.EventOFPPacketIn, MAIN_DISPATCHER)
  def _packet_in_handler(self, ev):

      msg = ev.msg
      datapath = msg.datapath
      ofproto = datapath.ofproto
      dst, src, _eth_type = struct.unpack_from('!6s6sH', buffer(msg.data),
          0)

      dpid = datapath.id
      pkt = Ether(msg.data)
      p = pkt[0]
      mac_src = pkt[0].src
      mac_dst = pkt[0].dst
      mac_dst_bin = dst
      mac_src_bin = src
```

1. **Extract important data from the OpenFlow message (listing 4.2)**:
   - The data path source address of the OpenFlow Switch that sent the OF message
   - Layer 2 header data
   - MAC source address
   - MAC destination address
2. **Check if the message is LLC or IPv6 (Listing 4.3)**: The CLC inspects the message and concludes that the message is in fact an LLC frame.
3. **Check the Binding Cache for an entry for the MN** since it's the first time the MN has been in the district, there are no entries for it in the CLC.
4. **Request information regarding the MN to the CRC** since it's the first time the MN has been in the CROWD network, there are no entries for it in the CRC.

5. **Classify the MN as new** since no Intradistrict or Interdistrict mobility has been detected.
6. **Assign an IPv6 address prefix**: The CLC extracts a prefix from the prefix pool if there is one available. A DMMGW is also assigned to the MN.
7. **Add entry in BC** for the MN with the newly assigned prefix and DMMGW.
8. **Send Binding Update to CRC** so the CRC can insert a new entry for the new MN.
9. **Install route in DMMGW (Listing 4.4)** needed for reachability to and from the outside for the MN.
10. **Generate a Router Advertisement (Listing 4.5)**: The CLC generates a RA on behalf of the DMMGW and sends it to the MN encapsulated in an OF message. This is done before the MN has a chance to send a RS in order to increase the speed of the whole operation.

Listing 4.3: Check the protocol type

```
try:
            is_arp = (p.type == 0x0806)
            is_ip = (p.type == 0x0800)
            is_ipv6 = (p.type == 0x86DD)
```

Listing 4.4: Install route in DMMGW

```
nMessage = "ip -6 route add " + prefijo + "/" + str(mascara) + " dev
    "+IF_INT_GW
udpbind.sendto(nMessage, (UDP_IP, UDP_PORT))
```

Listing 4.5: Generate, encapsulate and send the Router Solicitacion to the MN

```
RA_pk = self._build_RA_scapy(1, mac_dst.rstrip('\0'), mac_src.rstrip('\0'),
    prefijo, mascara)
actions = [datapath.ofproto_parser.OFPActionOutput(1, 0)]
buffer_id = 0xffffffff
in_port = datapath.ofproto.OFPP_LOCAL
message = datapath.ofproto_parser.OFPPacketOut(datapath, buffer_id, in_port,
    actions, str(RA_pk))
datapath.send_msg(message)
```

The MN is now ready to establish a connection with the CN. Traffic can flow towards and from the CN as soon as the OF rules are installed in the OF Switches, which will happen when the MN sends the first packet to the CN, expñlained in the next section.

4.2.2 Connection establishment

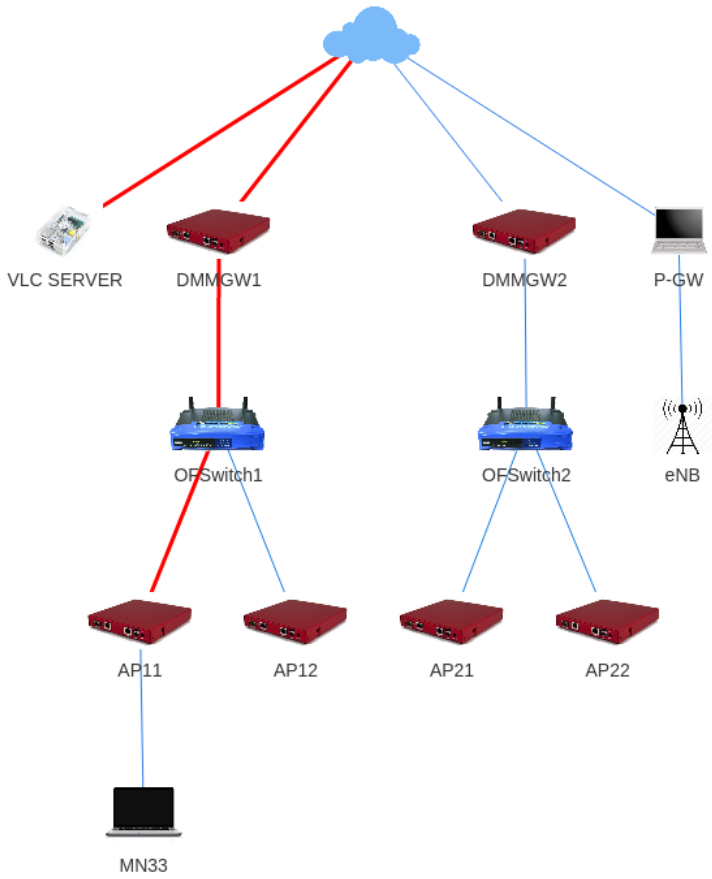


Figure 4.6: Connection establishment visualization

The MN is now completely associated with an AP in a CROWD district, ready to establish its first connection with the CN (represented in Figure 4.6). The MN will send an IPv6 packet to the CN, reaching the first OpenFlow Switch in the path. Since no rules have been set in the OF Table regarding traffic generated by that particular MN, it forwards the IPv6 packet to the CLC encapsulated in an OF message. An OpenFlow Packet-in Event will be raised in the CLC once it has received the message, executing the CLC Packet-in application following these steps:

1. **Extract important data from the OpenFlow message**:
   - The data path source address of the OpenFlow Switch that sent the OF message
   - Layer 2 header data
   - MAC source address
   - MAC destination address
2. **Check if the message is LLC or IPv6**: The CLC inspects the message and concludes that the message is an IPv6 packet.
3. **Check if the message was flowing downstream or upstream**: The traffic was generated by the MN, therefore it is flowing upstream.

4. **Check the BC for an entry for the MN** since the MN has completed the association process, there is an entry in the BC.

5. **Compute data path from the MN to the DMMGW assigned to the MN**: The CLC computes the shortest upstream path using the Dijkstra algorithm.

6. **Send OF rules to the OF Switches (Listing 4.6)**: The OF rules are requested to be installed in an OF message sent to the required OF Switches.

Listing 4.6: Install Upstream rules in OF nodes

```
output_port = self._install_path(path, datapath, msg, mac_src_bin,
    haddr_to_bin(macY), self.G_port_actual, self.G_in_ports_actual,
    src=str(mac_src), dst=str(macY), mn_id=mn_id)
output_port = self._install_path(path, datapath, msg, mac_src_bin,
    haddr_to_bin(FAKE_MAC_GW), self.G_port_actual, self.G_in_ports_actual,
    src=str(mac_src), dst=str(FAKE_MAC_GW), mn_id=mn_id),
    self.G_port_actual, self.G_in_ports_actual)
output_port = self._install_path(path, datapath, msg, mac_src_bin,
    haddr_to_bin("33:33:00:00:00:02"), self.G_port_actual,
    self.G_in_ports_actual, src=str(mac_src), dst="33:33:00:00:00:02",
    mn_id=mn_id)
```

Since the first packet the MN sent is consumed by the CLC, the CN will receive the second packet sent by the MN, but still cannot reach the MN. When the CN starts sending traffic towards the MN, it first reaches the DMMGW. The first course of action of the DMM is to send a Neighbor Solicitation to the MN which first reaches the OF Switch, encapsulating it in an OF message and forwarding it to the CLC. The CLC performs a similar operation as described above for the upstream path and reul creation. After this process concludes, the MN and the CN now have the means to exchange traffic between eachother.

4.2.3 Intradistrict mobility

The MN now decides to move to a new AP in the same district. The AP generates a LLC message which is forwarded by the OF Switch module running in the AP to the CLC of the same district via the Ryu framework encapsulated in an OpenFlow message, due to the OpenFlow Table not containing an entry instructing the OpenFlow Switch what to do with LLC frames (represented in Figure 4.7). An OpenFlow Packet-in Event will be raised in the CLC once it has received the message, executing the CLC Packet-in application following these steps:

1. **Extract important data from the OpenFlow message**:

   - The data path source address of the OpenFlow Switch that sent the OF message
   - Layer 2 header data
   - MAC source address

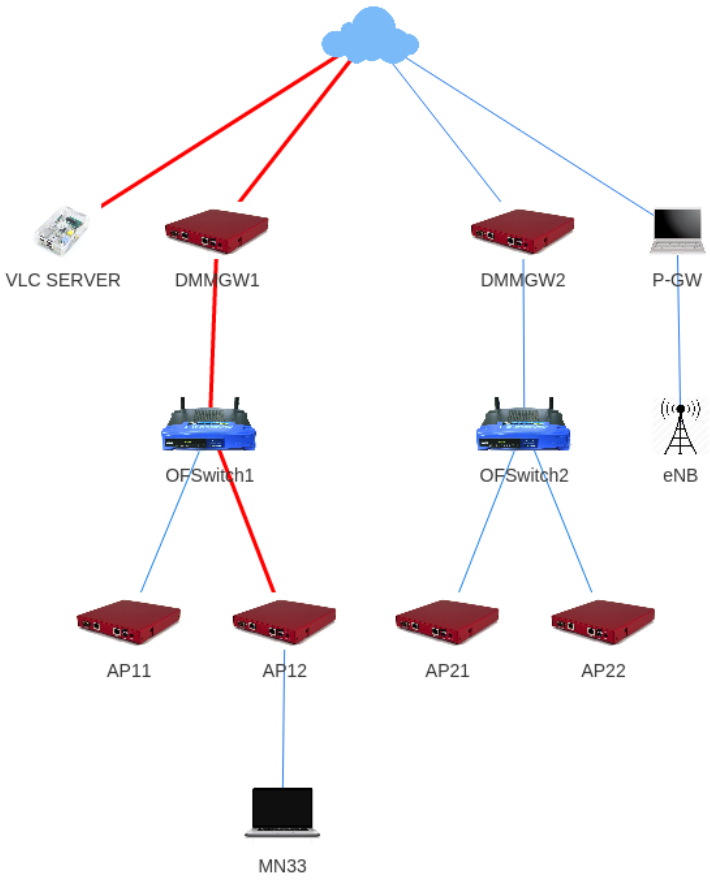


Figure 4.7: Intradistrict Handover visualization

- MAC destination address

2. **Check if the message is LLC or IPv6**: The CLC inspects the message and concludes that the message is in fact an LLC frame.

3. **Check the Binding Cache for an entry for the MN** since it's not the first time the MN has been in the district, there is an entry for it in the CLC.

4. **Classify the MN as old and performing an Intradistrict Handover** since the CLC has enough information to conclude that the MN is performing an Intradistrict Handover.

5. **Command OF Switches to delete old rules (Listing 4.7)** related to the MN. These rules no longer apply due the MN's new location.

6. **Send Binding Update to CRC** so the CRC can modify the entry it had for the MN updating it with the latest information.

7. **Generate a Router Advertisement**: The CLC generates a RA on behalf of the DMMGW and sends it to the MN encapsulated in an OF message. This is done before the MN has a chance to send a RS in order to increase the speed of the whole operation.

Listing 4.7: Delete existing rules for MN

```
self.remove_flow_entries_from_ip(path_to_remove, mac_src_bin, macY,
    src=str(mac_src), mn_id=mn_id)
self.remove_flow_entries_from_ip(path_to_remove, haddr_to_bin(macX), macY,
    src=str(macX), mn_id=mn_id)
self.remove_flow_entries_from_ip(path_to_remove,
    haddr_to_bin("33:33:00:00:00:02"), ofproto_v1_0.OFPFW_ALL,
    src="33:33:00:00:00:02", mn_id=mn_id)
```

The MN and the CN continue to send traffic between eachother, but the MN is unreachable for the CN, and the CN is unreachable for the MN. Due to the MN continuing to send traffic, when the process above concludes, the OF Switch module in the new AP will forward the first packet to the CLC encapsulated in an OF message, and will follow the processes described in section 4.2.2. This mobility is completely transparent to the user, so the MN and CN will continue almost without interruption to exchange traffic between eachother.

4.2.4 Interdistrict mobility

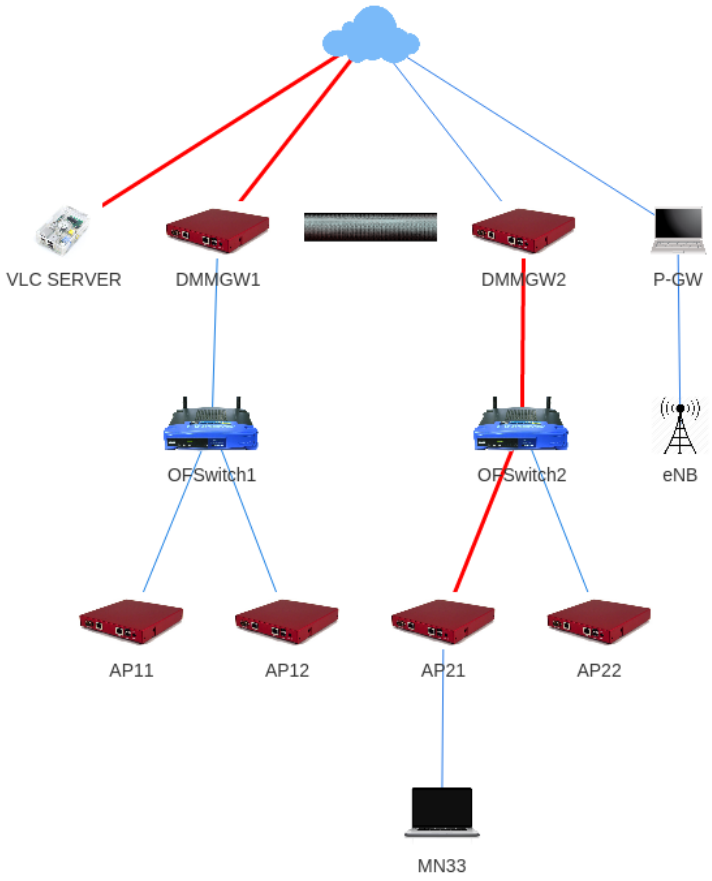


Figure 4.8: Interdistrict Handover visualization

The MN moves and enters a new CROWD District, associating with an AP. Like previous scenarios, the AP generates a LLC message which is forwarded by the OF Switch module running in the AP to the CLC of the same district via the Ryu framework encapsulated in an OpenFlow message, due to the OpenFlow Table not containing an entry instructing the OpenFlow Switch what to do with LLC frames. An OpenFlow Packet-in Event will be raised in the CLC once it has received the message, executing the CLC Packet-in application following these steps:

1. **Extract important data from the OpenFlow message**:
   - The data path source address of the OpenFlow Switch that sent the OF message
   - Layer 2 header data
   - MAC source address
   - MAC destination address
2. **Check if the message is LLC or IPv6**: The CLC inspects the message and concludes that the message is in fact an LLC frame.
3. **Check the Binding Cache for an entry for the MN** since it's the first time the MN has been in the district, there are no entries for it in the CLC.
4. **Request information regarding the MN to the CRC** since it has already been in a CROWD District, the CRC has an entry a for the MN in its BC.
5. **The CRC informs the old CLC** that the MN has moved out of the old district.
   a) **The CRC updates the BC entry** with the latest information.
   b) **The CRC informs the old CLC** that the MN has moved out of the old district.
   c) **The old CLC commands the OF Switches to delete routes** related to the MN that has moved.
   d) **The old CLC installs a tunnel in the old DMMGW** towards the MN's new DMMGW to redirect the traffic destined to the MN. The tunnel is created when the CLC receives a specific from the CRC, raised in the rectifier handler (Listing 4.8)
6. **Classify the MN as old and performing an Interdistrict Handover** since the CLC has enough information to conclude that the MN is performing an Interdistrict Handover.
7. **Install a tunnel in the new DMMGW** towards the MN's old DMMGW to receive the traffic directed to the MN.
8. **Generate a Router Advertisement**: The CLC generates a RA on behalf of the DMMGW and sends it to the MN encapsulated in an OF message. This is done before the MN has a chance to send a RS in order to increase the speed of the whole operation.

Listing 4.8: Tunnel creation between the DMMGWs

```
def rectifier():
            global MN_num
            s.bind((CLC_IP, CLC_PORT))
            udpbind = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
            s.listen(1)
            while 1:
                conn, addr = s.accept()
                while 1:
                    data = conn.recv(BUFFER_SIZE)
                    if not data: break

                    order, name, origin, destiny, prefix, prefixlen, mn,
                        direction = struct.unpack_from('!1s4s39s39s39si17s1s',
                            buffer(data))
                    mn_id = self._get_mn_id_from_BCE(mn)

                    mac_mcast = self._get_mcast_from_mac_in_BCE(mn)
                    mac_mcast_norm = self._get_mcast_from_mac(mn)

                    if (order == 'A'):
                        mensaje = "ip -6 rule add to " + prefix.rstrip('\0') +
                            "/" + str(prefixlen) +\
                                  " table myorg; ip -6 r a " +
                                     prefix.rstrip('\0') + "/" +
                                     str(prefixlen) + " dev " +\
                                  name.rstrip('\0') + " table myorg;"
```

4.2.5 Mobility to a non-CROWD network

The MN now decides to move to a non-CROWD district, attaching to a Point of Attachment not controlled by an OpenFlow-based SDN controller. Following the client-based DMM solution, the mobility is no longer completely transparent to the user, due to the MN having to intervene in the mobility process. As soon as the MN connects to the Point of Attachment, it executes a process that goes thorugh the following steps:

1. **The MN is assigned a IPv6 prefix**
2. **Execute the dhclient script to have an IPv4 address assigned**
3. **Communicate with the CRC to inform of the MN's new location**
   a) The CRC intalls an IPv4 tunnel between the DMMGW acting has the mobility anchor and the MN
4. **Receive the IPv4 address of the DMMGW acting has the mobility anchor**
5. **Create an IPv6 in IPv4 tunnel between the MN and the DMMGW associated**

6. **Create route to direct all traffic through tunnel**

4.3 Improvements

In this section we describe the improvements made to the project from the moment it was handed to us to continue the development.

4.3.1 Multi-node support

A natural progression in the project was to include multi-node support, meaning the provision of mobility support and connectivity to one or more mobile nodes. To provide multi-node support, the rules management in the CLCs had to be modified. Before the way rules where deleted when a MN performed a Intradistrict handover, was by deleting every rule with source the MAC address of the DMMGW. While working correctly when providing mobility support for only one mobile node, when more than one mobile nodes are connected to a CROWD District, by deleting all rules with source the DMMGW, it will delete all downstream rules related to all nodes in that district. To solve this problem, we had to modify the rules to be deleted, by telling the OpenFlow Switches to delete rules related to the MN's mac address.

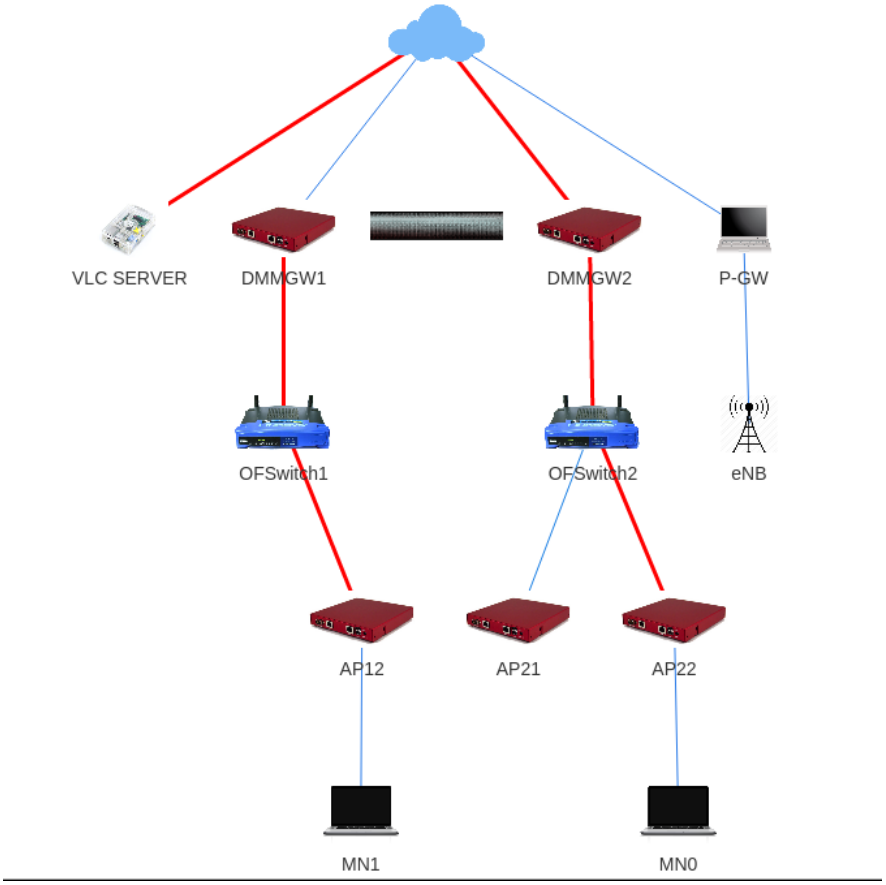


Figure 4.9: Multi-node support

4.3.2 Mobile node Whitelist

In order to have a greater control over the MN's that join the district, we have implemented a list of MN's allowed to attach. This is done by the CLC internally cross-referencing the list with the MN's MAC address. If the MN's MAC address is contained in the white list, it is allowed to attach to the network. If not, the MN is not granted access. This is done using a whitelist filter implemented as shown in Listing 4.9.

Listing 4.9: Whitelist filter

```
if (((mac_src != dmmgw_mac and self._in_whitelist(mac_src)) and
    self._is_whitelist_activated()) or mac_src == dmmgw_mac):
```

4.3.3 Resource Detection

An API added to the CLC is Resource Detection: when an OpenFlow Switch has booted, the CLC is able to detect it and use this information as the network administrator desires. To achieve this, the CLC application has been adapted to detect Hello messages sent by all the OF Switches at the start of the OF Switch module. This is useful becuase it is vital for the CLC to have real-time knowledge of the network. The implementation uses a Hello message handler, that detects and processes the Hello messages as seen in Listing 4.10, updating the OF ports (Listing 4.11). The OF ports are extracted from the CLC Config file saved in a Dictionary data structure, showed in Listing 4.12, that represent the path dictionary used by the Djikstra Algorithm with cost 1. The Dijkstra Algorith is used for calculating the shortest path between the MN and the DMMGW. Figure 4.11 shows a graphical representation of the CLCs detecting the APs connecting to the network:

Listing 4.10: Hello message handler

```
@set_ev_cls(ofp_event.EventOFPHello, HANDSHAKE_DISPATCHER)
    def _hello_handler(self, ev):
        msg = ev.msg
        datapath = msg.datapath

        self.update_G_port()
```

Listing 4.11: OpenFlow Port Update

```
def update_G_port (self):

        AP_21_true = 0
        AP_22_true = 0

        for i in range(len(self.node_info)):
            if (self.node_info[i] == AP1_IP):
                AP_21_true = 1
            elif (self.node_info[i] == AP2_IP):
                AP_22_true = 1
```

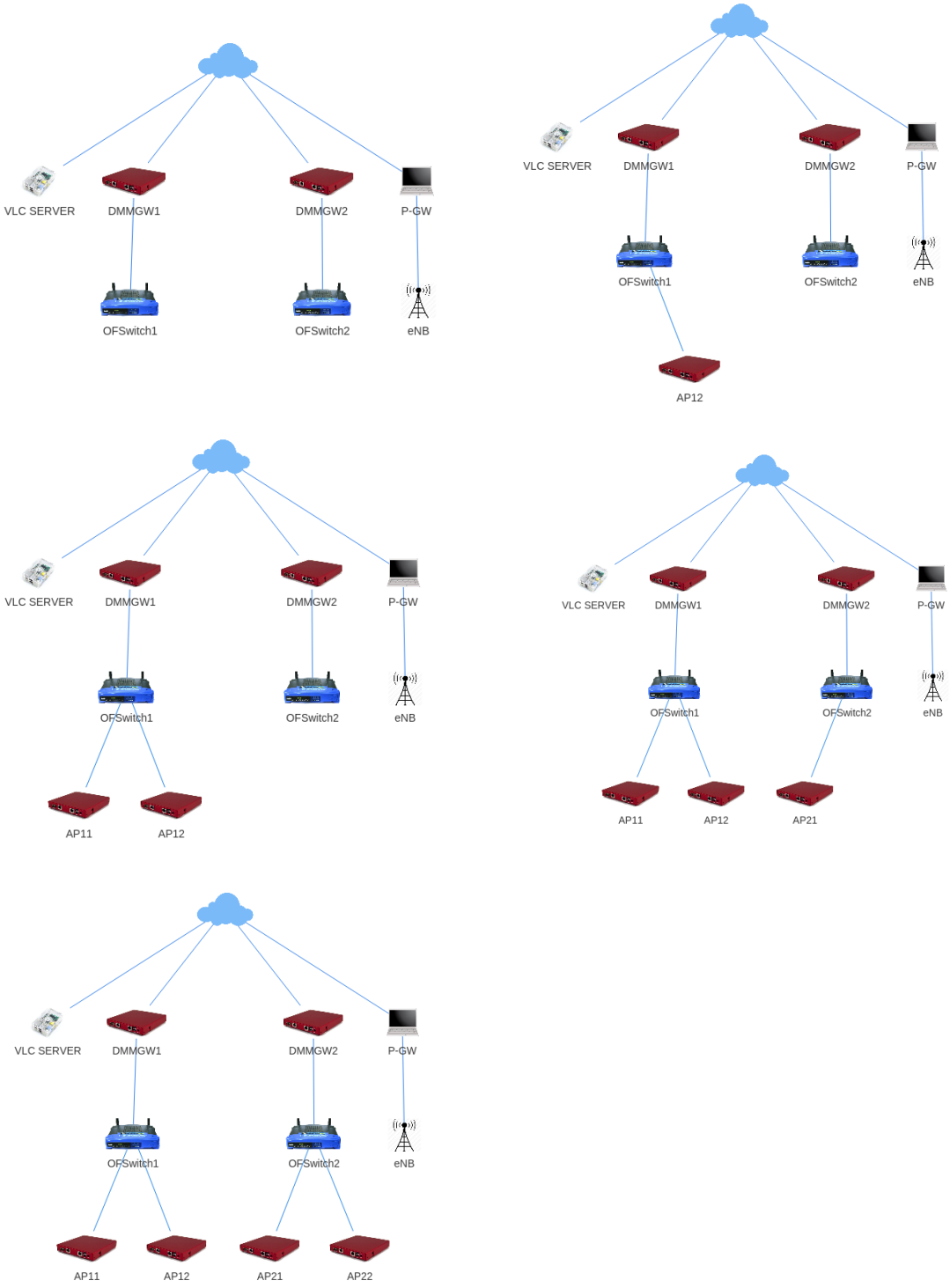

VLC SERVER
DMMGW1
DMMGW2
P-GW
OFSwitch1
OFSwitch2
eNB
AP12
AP11
AP21
AP22

Figure 4.10

```
        if (AP_21_true==1 and AP_22_true==1):
            #print blue("AP21 and AP22 are connected")
            self.G_actual = self.G_AP1_AP2
            self.G_port_actual = self.G_port_AP1_AP2
            self.G_in_ports_actual = self.G_in_ports_AP1_AP2
            self.G6_actual = self.G6_AP1_AP2
            self.G6_port_actual = self.G6_port_AP1_AP2
            self.G6_in_ports_actual = self.G6_in_ports_AP1_AP2
        elif (AP_21_true==1 and AP_22_true==0):
            #print blue("Only AP21 is connected")
            self.G_actual = self.G_AP1
            self.G_port_actual = self.G_port_AP1
            self.G_in_ports_actual = self.G_in_ports_AP1
            self.G6_actual = self.G6_AP1
            self.G6_port_actual = self.G6_port_AP1
            self.G6_in_ports_actual = self.G6_in_ports_AP1
        elif (AP_21_true==0 and AP_22_true==1):
            #print blue("Only AP22 is connected")
            self.G_actual = self.G_AP2
            self.G_port_actual = self.G_port_AP2
            self.G_in_ports_actual = self.G_in_ports_AP2
            self.G6_actual = self.G6_AP2
            self.G6_port_actual = self.G6_port_AP2
            self.G6_in_ports_actual = self.G6_in_ports_AP2
```

Listing 4.12: Port distribution dictionaries defined in the Config file of one CLC

```
G = {AP1_IP:{LSW_IP:1}, AP2_IP:{LSW_IP:1}, LSW_IP:{AP1_IP:1, AP2_IP:1,
    IP4_GW1:1}, IP4_GW1:{LSW_IP:1}}

G_AP1 = {AP1_IP:{LSW_IP:1}, LSW_IP:{AP1_IP:1, IP4_GW1:1}, IP4_GW1:{LSW_IP:1}}
G_AP2 = {AP2_IP:{LSW_IP:1}, LSW_IP:{AP2_IP:1, IP4_GW1:1}, IP4_GW1:{LSW_IP:1}}
G_AP1_AP2 = {AP1_IP:{LSW_IP:1}, AP2_IP:{LSW_IP:1}, LSW_IP:{AP1_IP:1,
    AP2_IP:1, IP4_GW1:1}, IP4_GW1:{LSW_IP:1}}

## Outgoing ports from current node
G_port = {AP1_IP:{LSW_IP:3}, AP2_IP:{LSW_IP:2}, LSW_IP:{AP1_IP:4, AP2_IP:2,
    IP4_GW1:1}, IP4_GW1:{LSW_IP:-1}}

G_port_AP1 = {AP1_IP:{LSW_IP:3}, LSW_IP:{AP1_IP:4, IP4_GW1:1},
    IP4_GW1:{LSW_IP:-1}}
G_port_AP2 = {AP2_IP:{LSW_IP:2}, LSW_IP:{AP2_IP:2, IP4_GW1:1},
    IP4_GW1:{LSW_IP:-1}}
G_port_AP1_AP2 = {AP1_IP:{LSW_IP:3}, AP2_IP:{LSW_IP:2}, LSW_IP:{AP1_IP:4,
    AP2_IP:2, IP4_GW1:1}, IP4_GW1:{LSW_IP:-1}}

## Incoming ports from current node
G_in_ports = {AP1_IP:{LSW_IP:3}, AP2_IP:{LSW_IP:2}, LSW_IP:{AP1_IP:4,
    AP2_IP:2, IP4_GW1:1}, IP4_GW1:{LSW_IP:1}}
```

```
G_in_ports_AP1 = {AP1_IP:{LSW_IP:3}, LSW_IP:{AP1_IP:4, IP4_GW1:1},
    IP4_GW1:{LSW_IP:1}}
G_in_ports_AP2 = {AP2_IP:{LSW_IP:2}, LSW_IP:{AP2_IP:2, IP4_GW1:1},
    IP4_GW1:{LSW_IP:1}}
G_in_ports_AP1_AP2 = {AP1_IP:{LSW_IP:3}, AP2_IP:{LSW_IP:2},
    LSW_IP:{AP1_IP:4, AP2_IP:2, IP4_GW1:1}, IP4_GW1:{LSW_IP:1}}

## IPv6 dictionary paths
G6 = {AP1_IP:{LSW_IP:1}, AP2_IP:{LSW_IP:1}, LSW_IP:{AP1_IP:1, AP2_IP:1,
    IP6_GW1:1}, IP6_GW1:{LSW_IP:1}}

G6_AP1 = {AP1_IP:{LSW_IP:1}, LSW_IP:{AP1_IP:1, IP4_GW1:1},
    IP4_GW1:{LSW_IP:1}}
G6_AP2 = {AP2_IP:{LSW_IP:1}, LSW_IP:{AP2_IP:1, IP4_GW1:1},
    IP4_GW1:{LSW_IP:1}}
G6_AP1_AP2 = {AP1_IP:{LSW_IP:1}, AP2_IP:{LSW_IP:1}, LSW_IP:{AP1_IP:1,
    AP2_IP:1, IP4_GW1:1}, IP4_GW1:{LSW_IP:1}}

## Outgoing ports from current node
G6_port = {AP1_IP:{LSW_IP:3}, AP2_IP:{LSW_IP:2}, LSW_IP:{AP1_IP:4, AP2_IP:2,
    IP6_GW1:1}, IP6_GW1:{LSW_IP:-1}}

G6_port_AP1 = {AP1_IP:{LSW_IP:3}, LSW_IP:{AP1_IP:4, IP6_GW1:1},
    IP6_GW1:{LSW_IP:-1}}
G6_port_AP2 = {AP2_IP:{LSW_IP:2}, LSW_IP:{AP2_IP:2, IP6_GW1:1},
    IP6_GW1:{LSW_IP:-1}}
G6_port_AP1_AP2 = {AP1_IP:{LSW_IP:3}, AP2_IP:{LSW_IP:2}, LSW_IP:{AP1_IP:4,
    AP2_IP:2, IP6_GW1:1}, IP6_GW1:{LSW_IP:-1}}

## Incoming ports from current node
G6_in_ports = {AP1_IP:{LSW_IP:3}, AP2_IP:{LSW_IP:2}, LSW_IP:{AP1_IP:4,
    AP2_IP:2, IP6_GW1:1}, IP6_GW1:{LSW_IP:1}}

G6_in_ports_AP1 = {AP1_IP:{LSW_IP:3}, LSW_IP:{AP1_IP:4, IP6_GW1:1},
    IP6_GW1:{LSW_IP:1}}
G6_in_ports_AP2 = {AP2_IP:{LSW_IP:2}, LSW_IP:{AP2_IP:2, IP6_GW1:1},
    IP6_GW1:{LSW_IP:1}}
G6_in_ports_AP1_AP2 = {AP1_IP:{LSW_IP:3}, AP2_IP:{LSW_IP:2},
    LSW_IP:{AP1_IP:4, AP2_IP:2, IP6_GW1:1}, IP6_GW1:{LSW_IP:1}}
```

4.3.4 Video streaming capability

A major improvement was to provide non-interrupted video streaming while mobility is being performed by the MN. The issue the project had previously was that mobility support couldn't be offered when performing Interdistrict Handovers, unless the MN sent a packet upstream first when reaching the new AP. The problem was that unless an upstream packet had been processed, the CLC would not be able to identify the destination MN given the destination IPv6 address when processing the Neigbour Solicitation message sent by the DMMGW if the destination address is a Link Local

address not formed using the MN's MAC address. This has been fixed by finding out how to extract the target IPv6 address from the Neigbour Solicitation message and extracting the prefix. Due to the prefix being in the BCE relative to the MN, the CLC can now identify the MN.

4.3.5 Infrastructure on Demand

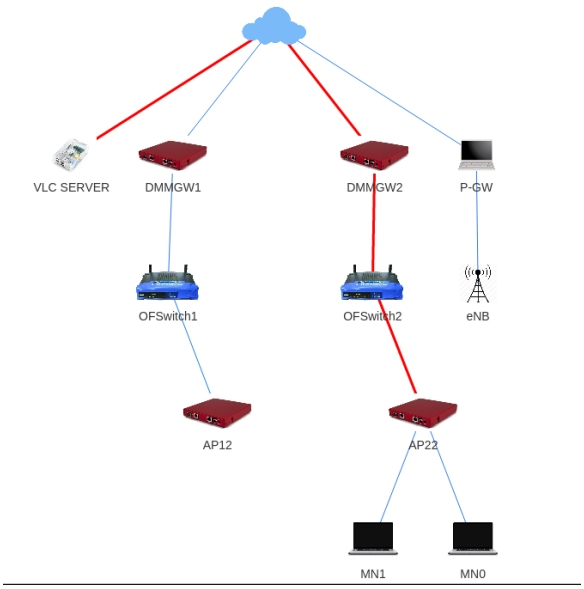
An important improvement was to add Infrastructure on Demand (IoD) capabilities, that allowes the CLC to activate or deactivate the APs. The initial state of the topology when starting a demo is one AP of each district is switched off, and the remaining two are switched on. The CLC is aware of the APs on and off thanks to the Resource Detection API explained before in section 4.3.3. If two MNs associate with the same AP, and the other AP is switched off, the CLC will switch the AP on by activating a port on the APC powerstrip via a python API. Once the AP is switched on, it will send a HELLO to the CLC that know knows the AP is on and ready. The CLC uses a command on the AP with connected APs via SSH that forces one of the MNs to perform a handover to the newly switched on AP. The IoD mechanism is performed only if a flag is set to true in the CLC config file, as can be seen in Listing 4.13. In Figure 4.11 we show a graphical view of the process: (a) only one MN is attached, (b) second MN attaches, (c) CLC switches on new AP, (d) CLC forces one of the MNs to perform a handover to the newly switched on AP.

Listing 4.13: Infrastructure on Demand

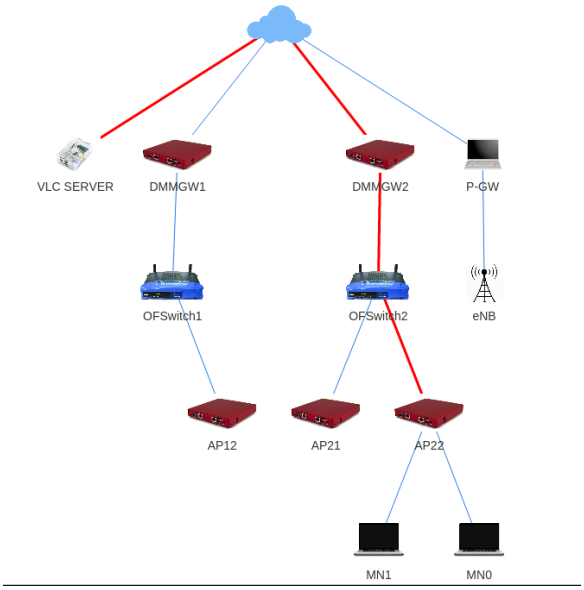
```
@set_ev_cls(ofp_event.EventOFPHello, HANDSHAKE_DISPATCHER)
    def _hello_handler(self, ev):
        msg = ev.msg
        datapath = msg.datapath

        self.update_G_port()

        if (datapath.address[0]==AP1_IP and self.ap_info[1] > 1 and
            APC_ENABLED==1):
            self.handover(2)
        elif (datapath.address[0]==AP2_IP and self.ap_info[0] > 1 and
            APC_ENABLED==1):
            self.handover(1)
```

4.3.6 Error control

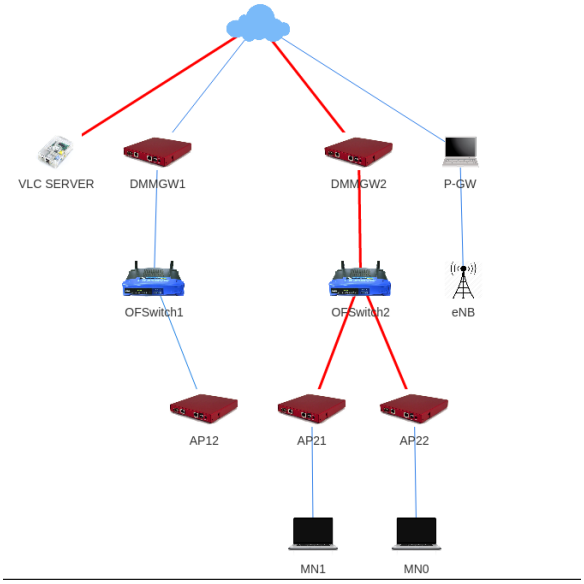
In some occasions, frequent enough that it needed dealing with, when the MN performs a Intradistrict handover, the CLC receives unvalid information due to OF switches having rules that haven't completed their rule deletion process, leading to the wrong rules being installed. The CLC now has the capacity to detect when this happens and avoid installing the wrong rules. Since no new rules have been installed, the next packet belonging to the same family will reach the CLC, and the packet misfired again, the process will be repeated.



(a)



(b)



(c)

Figure 4.11

4.4 Integration

In this section we describe the technologies that have been integrated with the implementation of the solution proposed since the project was handed to us.

4.4.1 LTE simulation

A major part of this final project was integrating LTE capabilities with the system.e have three laptops, two representing the UE and one the enB. From the two laptops representing the UE, one runs the GUI used for simulating handovers and the other runs the UE side of the ns-3 program (Chapter 2.3.1). The eNB runs the eNB side of the ns-3 program, the S-GW and the PDN-GW. If using the ns-3 program for LTE simulation, the LTE connection and the data transmission is completely simulated. If we also use the NI PXI, then LTE connection and data transmission is performed over a real channel using real transceivers.

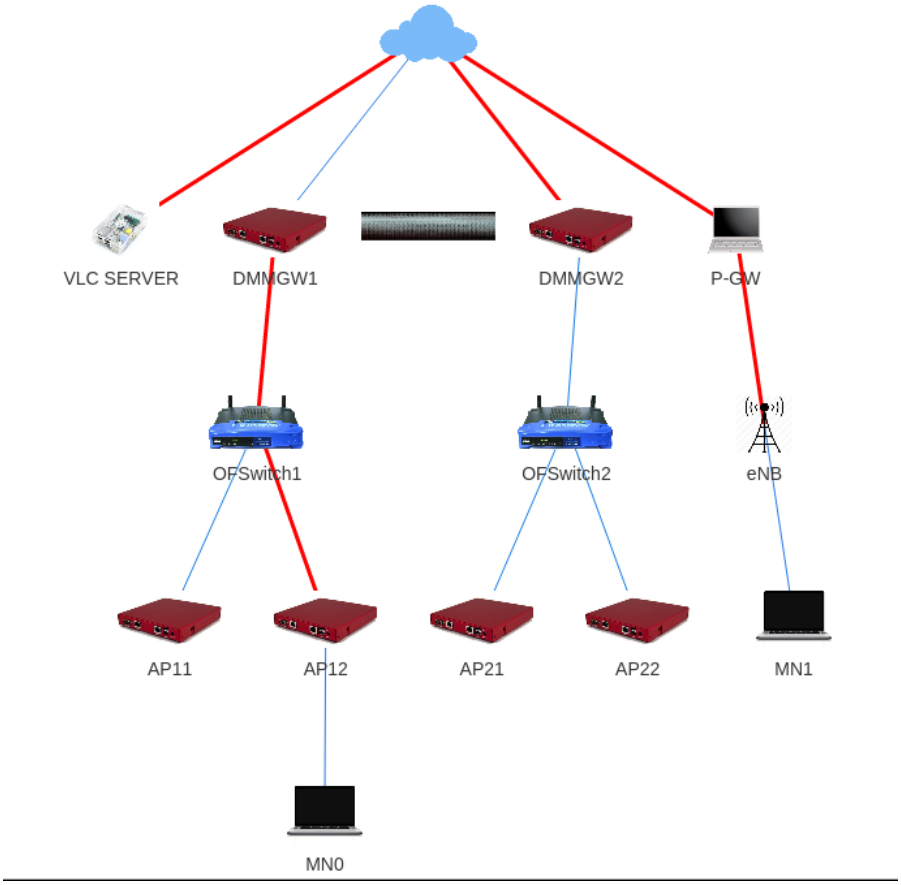


Figure 4.12: LTE Connection Representation

Listing 4.14: LTE connection setup

```
IMSI 1 RNTI 1 UeManager INITIAL_RANDOM_ACCESS --> CONNECTION_SETUP
IMSI 1 RNTI 1 UeManager CONNECTION_SETUP --> CONNECTED_NORMALLY
```

```
IMSI 1 RNTI 1 UeManager CONNECTED_NORMALLY --> CONNECTION_RECONFIGURATION
IMSI 1 RNTI 1 UeManager CONNECTION_RECONFIGURATION --> CONNECTED_NORMALLY
```

4.4.2 Video Server

The Video Server, implemented in a Raspberry Pi, runs a python UDP server (Listing 4.15) that accepts request messages for video streaming. When a MN wants to send a Video Start request message, a first NOP (No Operation) message must be sent because it will be consumed by the CLC for processing and will never reach the Video Server if there are no rules installed for the MN in the OF Switches. When the Video Start messages reaches the Video Server, it uses a VLC command to start streaming to the address that has sent the request using Real-time Transport Protocol (RTP), a network protocol for delivering audio and video over IP networks, to UDP port 1200 of the receiving MN. As soon as a video streaming request is sent, VLC is started on the MN awaiting the UDP video stream (Figure 4.13).

Figure 4.13: Video streamed in MN using VLC

Listing 4.15: Video Server

```
sock = socket.socket(socket.AF_INET6, socket.SOCK_DGRAM)
server_address = ('', 33333)
sock.bind(server_address)

while (True):

    data, address = sock.recvfrom(4096)
```

```
print "RECEIVED MESSAGE"

if (str(data)=="vlc_start"):
    print "Starting VLC Video Streaming"
    command = "cvlc ./big_buck_bunny_480p_surround-fix.avi --sout
        \"#rtp{dst=" + str(address[0]) + ",port=1200,mux=ts,ttl=3}\"
        :sout-keep :sout-standard-access=\"caching=1000\""
    thread.start_new_thread(os.system, (command,))

elif (str(data)=="vlc_stop"):
    print "Terminating VLC"
    subprocess.Popen(["pkill", "vlc"], stdout=subprocess.PIPE)
    subprocess.Popen(["pkill", "cvlc"], stdout=subprocess.PIPE)

else:
    print str(data)
    print "Message not recognized"
```

4.4.3 MaxiNet

MaxiNet is a Dynamic Backhaul Network emulation extendtion of the Mininet environment, created to span the emulation across several physical machines to allow the emulation of very large software-defined networks. As a part of this final project, we successfully integrated our system with a MaxiNet implementation acting as a black-box, to validate the reachibility of the Video Server through a unknown SDN network. A visual representation can be found in Figure 4.14

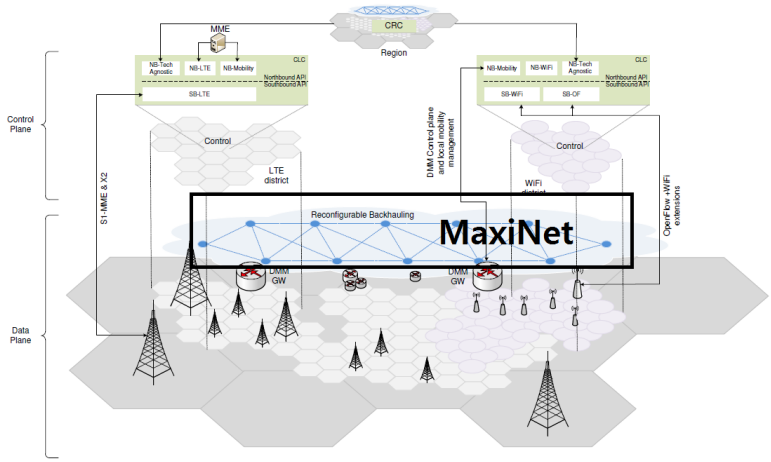


Figure 4.14: MaxiNet Integation. Adapted from "An SDN-based Network Arquitecture for Extremely Dense Wireless Networks,"

4.4.4 Visualization

A fundamental part of this project is the integration of real-time visualization of the network and its state. This has been achieved by developing a Web Site with its respective Web Server, that get the network information from the CLCs using an API. This has the capacity of illustrating:

- **Resource detection** (Section 4.3.3)
- **MN attachment**
- **Datapath installation**
- **Mobility**
- **Tunnel creation/deletion**
- **Infrastructure on Demand** (Section 4.3.5)

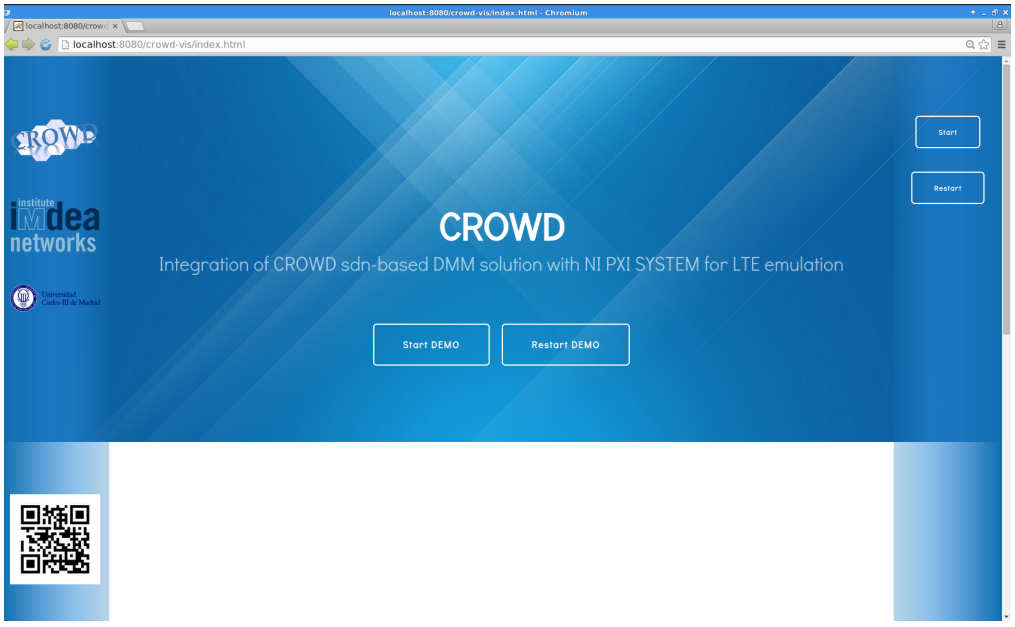


Figure 4.15: Web Site for Network Visualization

To view the MN clearly and simplify the backend-frontend communication, we have introduced unique identifiers, one assigned for each mobile node that connects to a CROWD district. As soon as a MN connects to an AP controlled by a CLC, an identifier is assigned by the CLC. The identifier will be included in the BC of the CLC and the BC in the CRC.

Backend

The backend Web Server is implemented using Java Servlets and Apache HTTP Server. When the CLC installs a data path, or creates a tunnel, it creates a message in JSON format and sends it via HTTP POST to the Web Server. The format of the JSON messages depends on the Operation Type field. Here are the formats:

- **Operation: 0**: Sent when a MN attaches to an AP for the first time
  - Operation: 0
  - MN ID: The unique identifier of the MN
  - Position: Position of attachment
- **Operation: 1**: Sent when a MN performs a handover
  - Operation: 1
  - MN ID: The unique identifier of the MN
  - Position: New position of the MN
- **Operation: 2**: Sent when a datapath has been installed or deleted
  - Operation: 2
  - Pathcode: code number of the path
- **Operation: 3**: Sent when a tunnel has been created or deleted
  - Operation: 3
  - Tunnel ON: 1 if the tunnel icon should be activated, 2 if it should be deactivated
- **Operation: 4**: Sent when an AP has been switched on or off
  - Operation: 4
  - AP ID: Identifier of the AP

The Server receives the message and saves it in a file containing a JSON Array called Message Queue, acting as a First In First Out (FIFO) queue for messages to be sent to the Web Site. The Web Site will asynchronously and periodically request the Server for new information. The request is processed by the Server by checking the Message Queue for the next piece of information to be sent. If the Message Queue is not empty, it will extract the next message and send it to the Web Site in the original JSON format. When the Web Site receives the JSON message, it first extracts the operation field to check the operation option. Then, depending on the operation option, will extract the remainder of the data and perform the corresponding operations.

4.5 Debugging

From the start of the project, every step forward towards improving the functionality of the system, optimising its mechanisms and integrating it with a variety of technologies has required exhaustive testing and debugging of the code.

CHAPTER 5

Design Alternatives

5.1 CROWD Controller hardware

The CROWD Local Controllers and the CROWD Regional Controllers were originally Linux personal computers. Raspberry Pi's were tested to see if they could replace the Linux PC's, in order to implement the Controller applications in a much more mobile platform. Due to not being powerful enough, they were finally replaced by single-board computers called Minnowboard. The Minnowboard offered the possibility of having its higher processing resources dedicated exclusively to the Controller functions. Being much smaller in size, they are much easier to transport and to set up. On top of this, they offered HDMI capabilities, contrary to their pc counterpart. More information about the Minnowboard can be found in Chapter 4.1.1.

5.2 LTE emulation tool

LTE emulation could be perfomed with either ns-3, a discrete event network simulator used to create an open simulation environment for computer networking research, or with a combination with ns3 and NI PXIe systems (a detailed description of each solution can be found in Chapter 2.3.1. While ns-3 + NI PXIe systems offered the most realistic results, due to using real non-emulated transceivers for data transmission and reception, the NI PXIe systems are very expensive, hard to transport, hard to set-up, and not always available to use.

5.3 SDN Framework Ryu

The choice of using Ryu over OpenDaylight as an SDN Framework was made because of the much better and more accessible documentation, and because of the fact that Ryu supports python programming language.

5.4 Infrastructure on Demand switch off mechanism

The original option at hand was to switch the APs directly with a poweroff command, but this has the problem of not being able to switch them back on. To solve this issue, we used an APC powerstrip with IP connectivity that can dynamically switch the APs on and off remotely via commands. This mechanism used for switching off Access Points has the drawback of doing so in an aggressive manner, meaning the Access Points are switched off directly cutting the electric power. This is a solution limitation that does not have a major impact, so an alternative has not been taken into consideration.

5.5 Access point hardware

The options at hand were either using Linksys routers for the Access Points, or using an Alix board. The decision of choosing the Alix board instead of the Lynksis router was weighing the limitation of the Linksys router which is that it can't run the linux distribution needed, against the drawback of the Alix board that is the long time it takes to start up when switched off.

5.6 OpenFlow Version

The OpenFlow version used is 1.0 due to the Lynksis routers not being compatible with higher verions. This version has many limitations and is very unstable. The reason behind choosing the Lynksis over the Alix PC (that can run higher versions of OpenFlow) like for the APs, is that the Linksys provide the number of ports needed (5, compared to the 3 ports the Alix PC has). A design alternative could have been including a usb to ethernet adapter, but we were limited to our budget. Another limitation of the Linksys over the Alix PC is the lack of vlan capabilities, so the datapath rules had to be installed in each node one by one.

CHAPTER 6

Experiments

Taking into consideration that this project is a *proof of concept*, in order to prove the correct functionality of the system and measure the performance, fully comprehensive experiments and tests have been carried out throughout the life cycle of the project. The experiments were implemented using bandwidth performance tools such as iperf and using Wireshark as a network protocol analyzer.

6.1 Initial Attachment

In this section we will show the time measurement results obtained when the MN associates to an AP and starts to generate traffic. Illustrated in the Figure 6.3 is a chart with the average of the times measured from when the MN attaches to the AP until it receives a reply packet from the DMMGW. First the LLC packet is processed by the CLC when the MN associates with the AP, then the AP sends a ICMPv6 packet to the DMMGW, processed by the CLC, after which will install the upstream rules in the OF nodes. Then the DMMGW receives the request and sends a reply and the downstream path is installed following an analogous process. The process can be found described with more detail in Chapter 4.2.1 and 4.2.2.

6.1.1 Attachment

The time consumed from the moment the MN attaches to an AP, to the moment it receives a RA is represented in Figure 6.1 as a cumulative distributed function (CDF).

6.1.2 Connection establishment

The time consumed from the moment the MN receives a RA and sends first packet to the DMMGW, to the moment it receives the first packet from the DMMGW is represented in Figure 6.2 as a CDF.

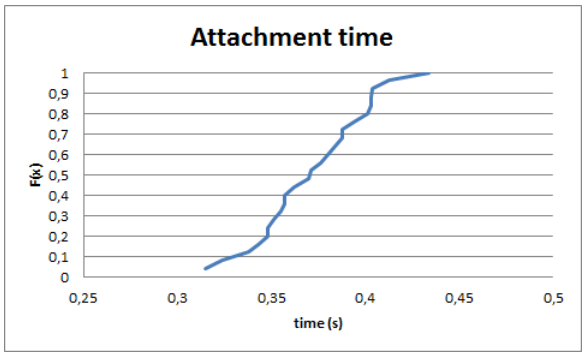


Figure 6.1

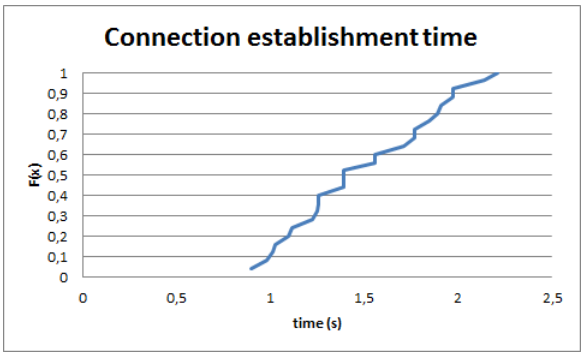


Figure 6.2

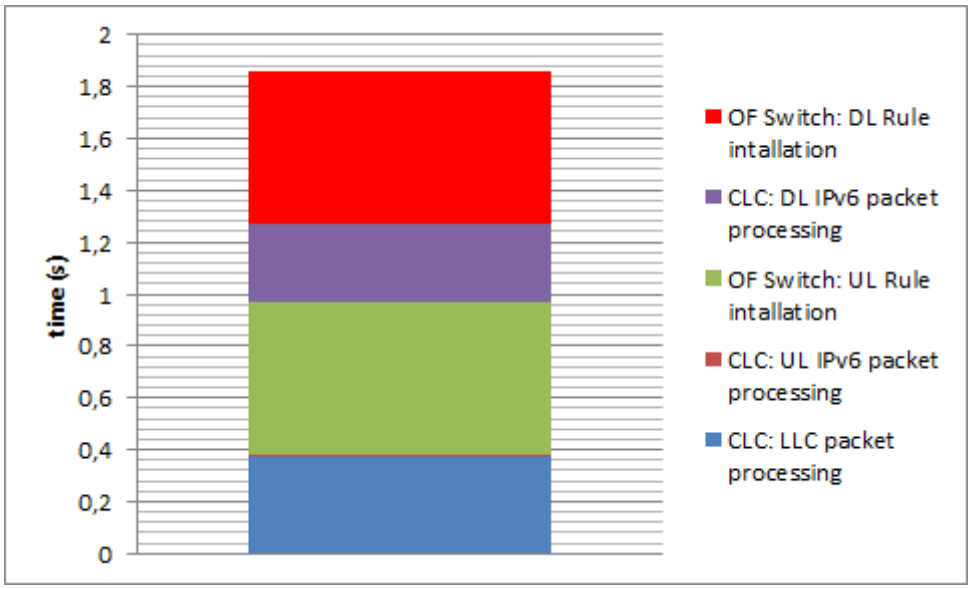


Figure 6.3

6.2 Handovers

In this section we focus on the experiments performed for mobility time measurements.

6.2.1 Intradistrict

The time measurement results obtained when the MN does a Intradistric handover is represented in the Figure 6.4. The processing involved consists in deleting the old paths installed in the OF nodes in the district related to the MN, and install new ones for the new datapath. The Figure 6.5 shows a chart with the average of the times measured when the LLC is processed and the OF downstream rules are created (the deletion of the OF paths is included in the OF rule creation processing time). We don't contemplate the upstream rules due to working in the context of video streaming. The process can be found described with more detail in Chapter 4.2.3

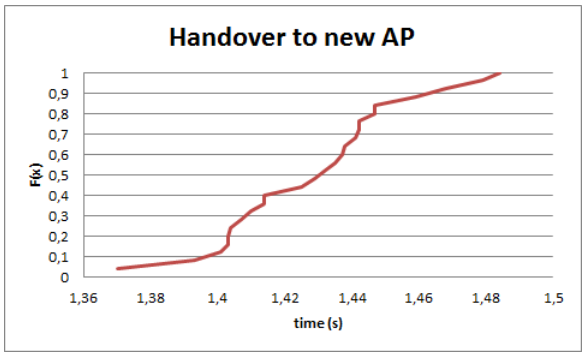


Figure 6.4

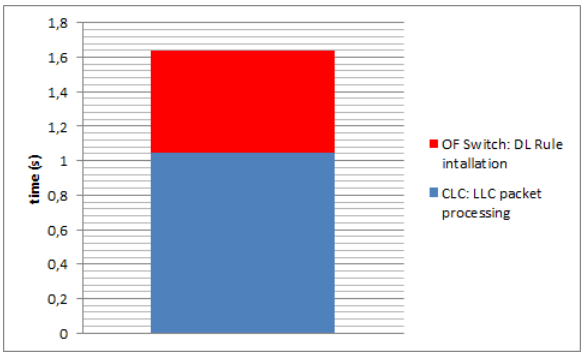


Figure 6.5

6.2.2 Interdistrict

The time measurement results obtained when the MN does a Intradistric handover are represented in the Figure both in the Figure 6.6 for the case when the MN performs a Interdistrict handover. The Figure 6.7 shows a chart with the average of the times measured when the LLC is processed, the creation of the tunnel between the DMMGWs, contemplating the time the CLCs and CRC take for processing, sending the command to the DMMGWs and the tunnel being created, and the OF downstream rules are created. We don't contemplate the upstream rules due to working in the context of video streaming. The process can be found described with more detail in Chapter 4.2.4

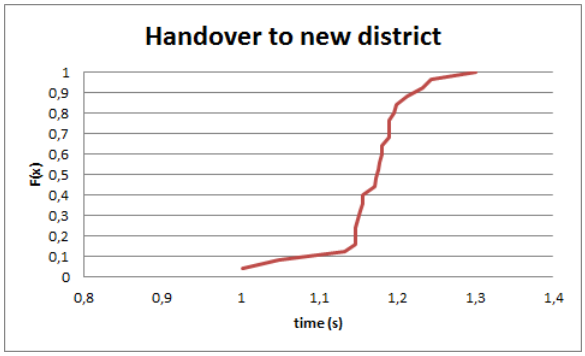


Figure 6.6

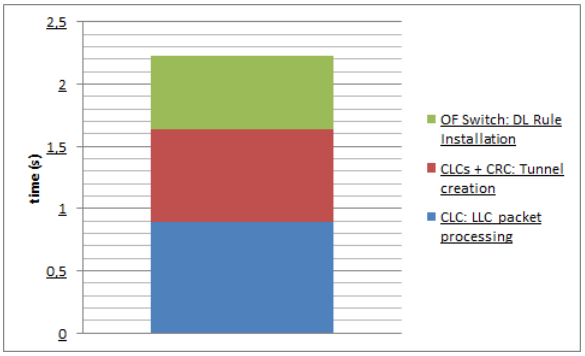


Figure 6.7

In Figure 6.8 we show a comparison between the Intradistrict and Interdistrict handover latencies. The reasoning behind the Intradistrict handover taking longer to complete than the Interdistrict handover, is that in an Intradistrict handover, rules have to be deleted first before the new rules can be installed. If a rule hasn't been deleted the CLC may not receive any packets, so no processing can done, therefore no new rules can be installed.

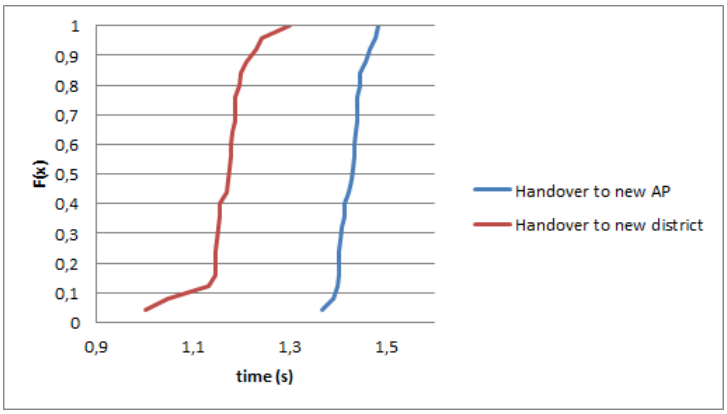


Figure 6.8

6.2.3 LTE

The time measurement results obtained when the MN does a handover to the LTE district is represented in the Figure 6.9. The Figure 6.10 shows a chart with the average of the times measured for the MN attachment process and the time taken to create the tunnel between the MN and the old DMMGW. The process can be found described with more detail in Chapter 4.2.5.

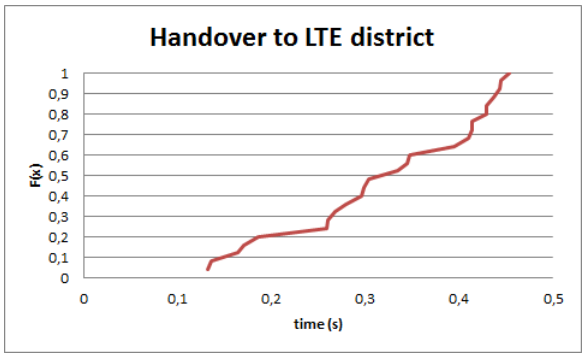


Figure 6.9

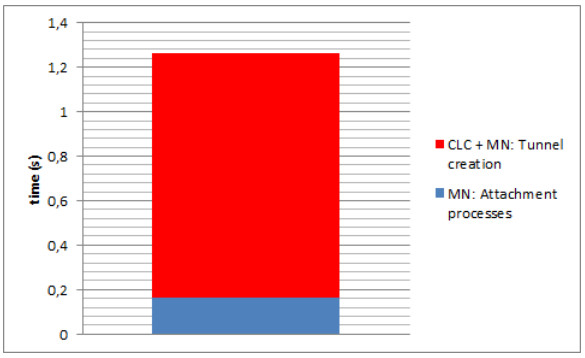


Figure 6.10

6.2.4 Summary of obtained results

In the Figure 6.11 we show a summary of the results obtained for the scenarios that have just been discussed.

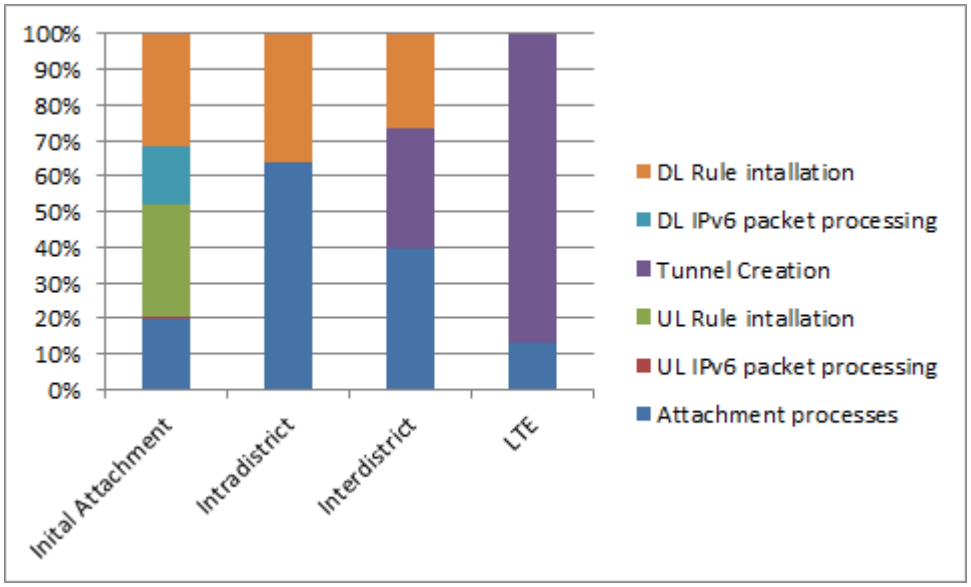


Figure 6.11

6.3 Infrastructure on Demand

The graph illustrated in Figure 6.12 represents the throughput of each of the MNs connected to one AP. At time 0, only one MN is attached to the AP, until a second MN attaches to the same AP. At this point the throuput of the first MN goes down and the the CLC switches on a new AP and forces one of the MNs to perform a handover to the newly switched on AP. It can be seen that the when the handover is performed, the sum of the throughputs is double the original.

6.4 Performance of the network components

In this section we detail the performance of the different network components contained in the architecture.

6.4.1 Initial attachment

In this section we analyse the performance of different network components during the first attachment of the MN. In the Figure 6.1 we show the performance of the CROWD Local Controller when processing LLC Frames, when computing the upstream data path and generating the ruled to be installed in the different OpenFlow nodes, and

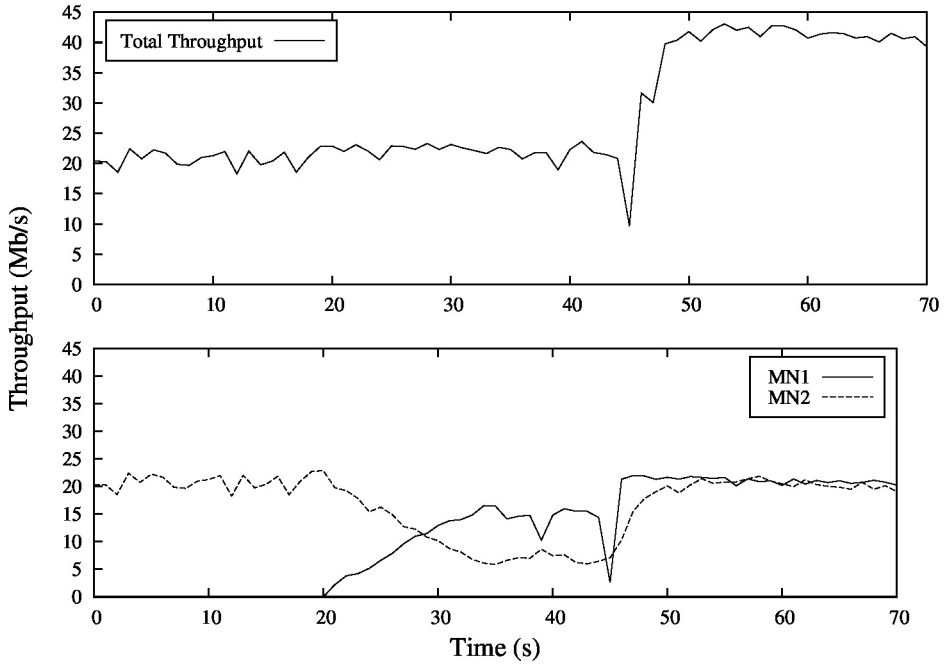


Figure 6.12: Throughput measurement with the RoD mechanism. Adapted from "An OpenFlow Architecture for Energy Aware Traffic Engineering in Mobile Networks"

when computing the downstream data path and generating the ruled to be installed in the different OpenFlow nodes.

6.4.2 Intradistrict handover

In this section we analyse the performance of different network components during an Intradistrict handover. In the Figure 6.2 we show the performance of the CROWD Local Controller when processing LLC Frames generated when the MN does an Intradistrict handover, and computing time for the downstream data path and generating the ruled to be installed in the different OpenFlow nodes.

6.4.3 Interdistrict handover

In this section we analyse the performance of diffent network components during an Interdistrict handover. In the Figure 6.3 we show the performance of the CROWD Local Controller when processing LLC Frames generated when the MN does an Intradistrict handover, the combined performance of the two CROWD Local Controllers and the CROWD Regional Controller to create the tunnel, combined performance of the two CROWD Local Controllers and the CROWD Regional Controller to delete the tunnel hen the MN performs an Interdistrict handover back to the the old Crowd District.

6.4.4 Non-CROWD Interdistrict handover

In this section we analyse the performance of diffent network components during a handover to the LTE district. In Figure 6.4 we show the performance of Mobile Node when attaching to the LTE district, the combined performance of the CROWD

LLC Packet Processing	
Average:	0,3704
Median:	0,3595
Min:	0,338
Max:	0,413
Percentile 25:	0,34875
Percentile 75:	0,39775

UL IPv6 Packet processing	
Average:	0,0087723
Median:	0,009295
Min:	0,00101
Max:	0,0124
Percentile 25:	0,0088775
Percentile 75:	0,0094425

DL IPv6 Packet processing	
Average:	0,3611
Median:	0,354
Min:	0,313
Max:	0,412
Percentile 25:	0,3425
Percentile 75:	0,38675

Table 6.1

LLC Packet Processing	
Average:	1,0188
Median:	1,01
Min:	0,998
Max:	1,047
Percentile 25:	0,999
Percentile 75:	1,04

DL IPv6 Packet processing	
Average:	0,01628
Median:	0,0145
Min:	0,0098
Max:	0,027
Percentile 25:	0,011
Percentile 75:	0,02175

Table 6.2

Local Controllers and the CROWD Regional rController to create the tunnel and the combined performance of the CROWD Local Controllers and the CROWD Regional Controller to create the tunnel when the MN performs a handover back to a CROWD district.

LLC Packet Processing	
Average:	0,9013
Median:	0,8315
Min:	0,676
Max:	0,143
Percentile 25:	0,72125
Percentile 75:	1,1305

Tunnel creation time	
Average:	0,7531
Median:	0,7525
Min:	0,745
Max:	0,764
Percentile 25:	0,74725
Percentile 75:	0,75825

Tunnel deletion time	
Average:	1,1113
Median:	1,1185
Min:	0,998
Max:	1,152
Percentile 25:	1,11425
Percentile 75:	1,12975

Table 6.3

MN Attachment process	
Average:	0,165
Median:	0,166
Min:	0,142
Max:	0,189
Percentile 25:	0,15
Percentile 75:	1,179

Tunnel creation time	
Average:	1,0988
Median:	1,0995
Min:	1,076
Max:	1,118
Percentile 25:	1,0905
Percentile 75:	1,10725

Tunnel deletion time	
Average:	0,3707
Median:	0,3695
Min:	0,362
Max:	0,38
Percentile 25:	0,36525
Percentile 75:	0,37725

Table 6.4

CHAPTER 7

Planning

7.1 Time planning

The scope for the development was limited by time constraints, therefore optimal planning was fundamental to maximize the value of this final project. A Gantt diagram has been used to summarize the planning, portrayed in Figure 7.1 and Figure 7.2.

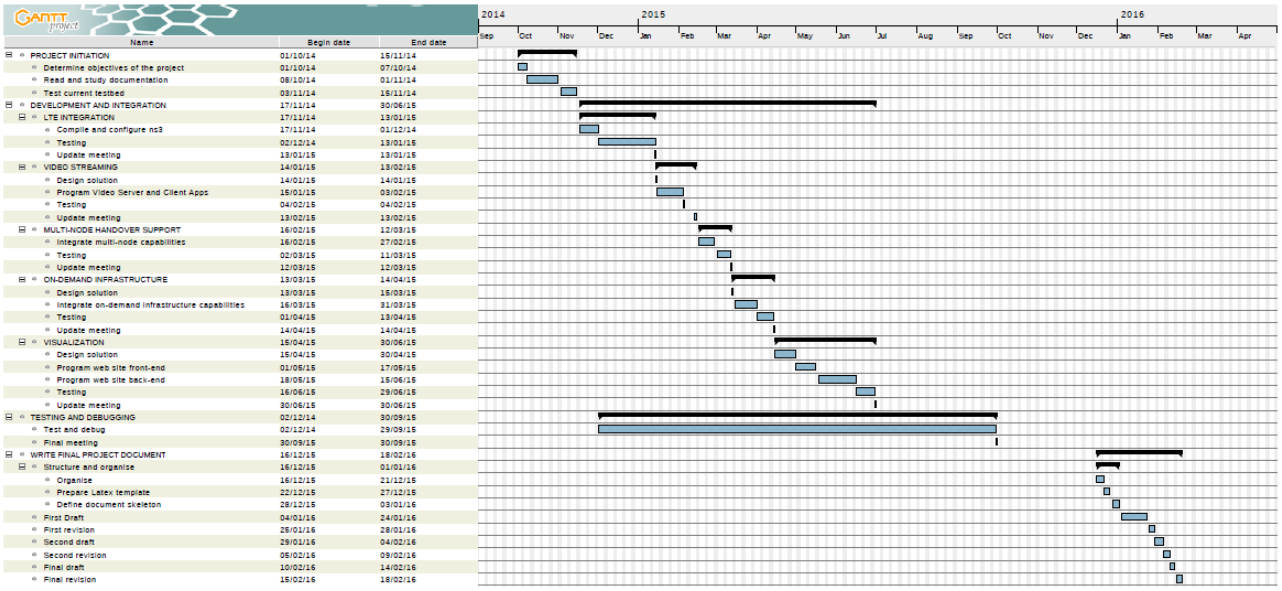
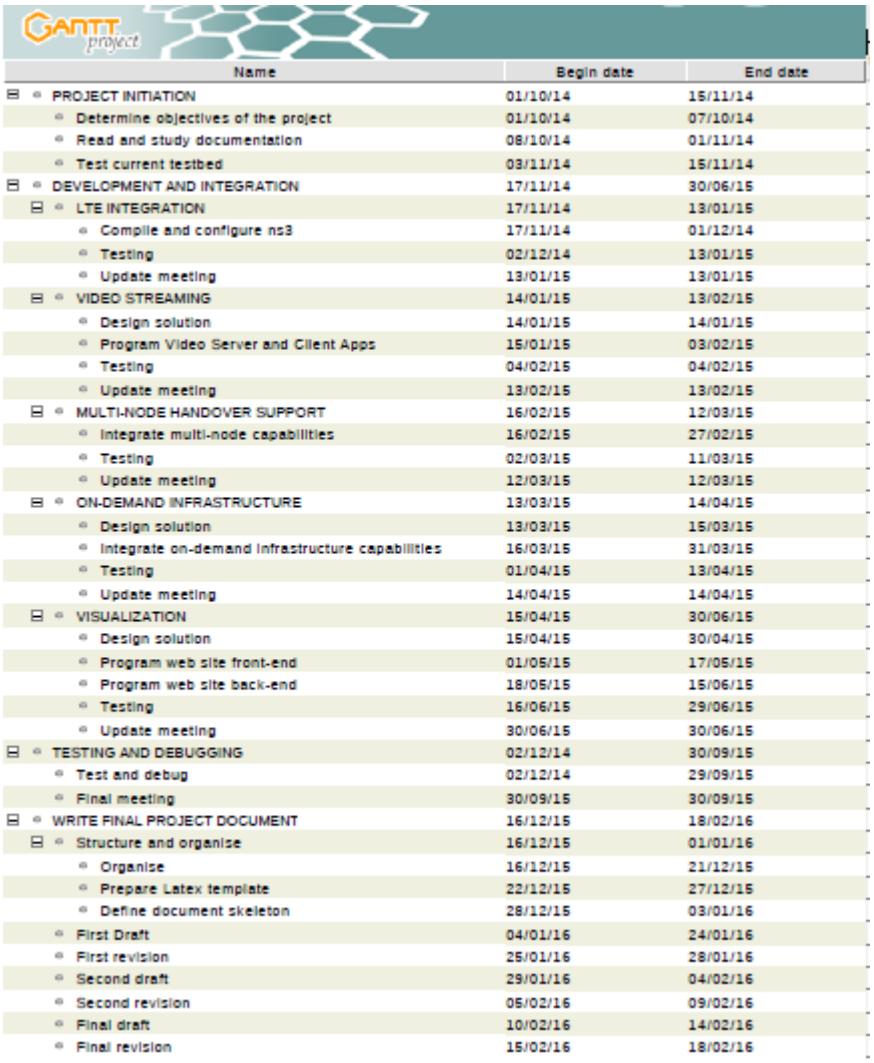


Figure 7.1: Gantt Chart

7.2 Cost Estimate

7.2.1 Human Resources

The following table shows an estimation of the cost of personnel resources used in the development of this project:



Name	Begin date	End date
PROJECT INITIATION	01/10/14	15/11/14
Determine objectives of the project	01/10/14	07/10/14
Read and study documentation	08/10/14	01/11/14
Test current testbed	03/11/14	15/11/14
DEVELOPMENT AND INTEGRATION	17/11/14	30/06/15
LTE INTEGRATION	17/11/14	13/01/15
Compile and configure ns3	17/11/14	01/12/14
Testing	02/12/14	13/01/15
Update meeting	13/01/15	13/01/15
VIDEO STREAMING	14/01/15	13/02/15
Design solution	14/01/15	14/01/15
Program Video Server and Client Apps	15/01/15	03/02/15
Testing	04/02/15	04/02/15
Update meeting	13/02/15	13/02/15
MULTI-NODE HANDOVER SUPPORT	16/02/15	12/03/15
Integrate multi-node capabilities	16/02/15	27/02/15
Testing	02/03/15	11/03/15
Update meeting	12/03/15	12/03/15
ON-DEMAND INFRASTRUCTURE	13/03/15	14/04/15
Design solution	13/03/15	15/03/15
Integrate on-demand infrastructure capabilities	16/03/15	31/03/15
Testing	01/04/15	13/04/15
Update meeting	14/04/15	14/04/15
VISUALIZATION	15/04/15	30/06/15
Design solution	15/04/15	30/04/15
Program web site front-end	01/05/15	17/05/15
Program web site back-end	18/05/15	15/06/15
Testing	16/06/15	29/06/15
Update meeting	30/06/15	30/06/15
TESTING AND DEBUGGING	02/12/14	30/09/15
Test and debug	02/12/14	29/09/15
Final meeting	30/09/15	30/09/15
WRITE FINAL PROJECT DOCUMENT	16/12/15	18/02/16
Structure and organise	16/12/15	01/01/16
Organise	16/12/15	21/12/15
Prepare Latex template	22/12/15	27/12/15
Define document skeleton	28/12/15	03/01/16
First Draft	04/01/16	24/01/16
First revision	25/01/16	28/01/16
Second draft	29/01/16	04/02/16
Second revision	05/02/16	09/02/16
Final draft	10/02/16	14/02/16
Final revision	15/02/16	18/02/16

Figure 7.2: Project Activities

Name	Category	Cost/Hour	Hours	Cost
Tutor	Senior Engineer	36	80	2880
PhD Student	Senior Engineer	36	200	7200
Grad Student	Engineer	20	1042	20875

Table 7.1: Human resources costs

7.2.2 Hardware Costs

The following table details the cost of the hardware used in this project:

Concept	Units	Cost/Unit	Cost
Alix 2d3	6	89,8	538,8
Linksys WRT54GL	2	50	100
Ethernet wire	40	2	80
D-Link 24-Port Switch	1	177	177
Netgear FS605NA 5-Port Switch	1	20	20
Raspberry Pi	1	50	50
MinnowBoard	3	100	300
Laptop	4	700	2800

Table 7.2: Hardware costs

7.2.3 Software Costs

The software used for the development of this project is open source, therefore it was free of cost.

7.2.4 Cost Summary

The following table shows a summary of the total costs spent for the development of this project:

Concept	Total Cost
Human resources	30955
Software	0
Hardware	4071,8

Table 7.3: Summary of total costs

The total cost of the project is **35026,8 euros**.

CHAPTER 8

Regulatory Framework

All the technologies and protocols used in this project listed below follow the corresponding specification:

- **OpenFlow 1.0**: All the OpenFlow nodes follows the OpenFlow 1.0 specification.
- **OpenFlow vSwitch 1.0**: The OpenFlow Switches follow the OpenFlow vSwitch 1.0 specification.
- **802.11**: All the OpenFlow nodes used in the project are IEEE 802.11 compliant.
- **802.11k**
- **802.11v**

Client and Network based solutions for Distributed Mobility Management are in the process of being standardized by the IETF DMM Working Group: draft-bernardos-dmm-cmip-00 for the Client Mobile IP (host) based solution and draft-bernardos-dmm-pmip-02 for the Proxy Mobile IP (network) based solution.

CHAPTER 9

Socioeconomic Environment

As explained in Chapter 1.1, the mobility solutions used today are not prepared for the upcoming 5G technologies and specifications, scalability is not efficient which means a great cost for the operators, as well as non-optimal existing mechanisms to deal with extremely dense networks. SDN and DMM have been proposed as high impact solutions to these problems.

The European Commission has commissioned a study on the "Implications of the emerging technologies Software-Defined Networking and Network Functions Virtualization on the future Telecommunications landscape" [22]. One of the objectives of the study is to explore the unknown area of technological socio-economic-regulatory impact of Software-Defined Networking. The study aims to:

- Identify the most likely deployment scenarios including associated timelines and possible migration paths from existing networking technologies
- Identify current and new usage scenarios of SDN-based networks including trends and possible new services
- Assess the impact of SDN on existing business models in the telecommunications sector and identify their innovation potential in terms of possible new business models with current and new stakeholders
- Assess the impact of SDN on existing business models in the telecommunications sector and identify their innovation potential in terms of possible new business models with current and new stakeholders
- Assess the general market/industrial potential for SDN.
- Position SDN within the current and future telecommunications regulatory framework and in relation to growth opportunities.

InterDigital Europe, the company's London-based 5G-focused research unit, has been chosen by the European Commission to conduct a 5G socioeconomic research project [28]. The research will directly inform the development and rollout of 5G in the European Union, and empower the region to stake an early claim as a global 5G leader. The study will produce an unprecedented framework, based on key socioeconomic data, for the European Commission's introduction of 5G infrastructure in

Europe. The project will engage key stakeholders across the wireless industry, as well as vertical markets and policy makers, to get a clear picture of 5G's economic impact and the possible innovation it will spur in areas such as health, transport, social services and commercial opportunities. Executed over the next year, the study will be completed by InterDigital in partnership with additional 5G experts including Real Wireless, Rethink Wireless, Tech4i2 and Trinity College, Dublin. The study will:

- Map out possible 5G scenarios in the region's economy and society to determine technological requirements and parameters for 5G implementation.
- Quantify rural, suburban and urban usage patterns and their impact on technology requirements
- Identify expected use cases for 5G technology, and their impact on technical parameters
- Analyze spectrum demand and technical spectrum issues, including quantitative needs and boundaries, and consider new band requirements and constraints

CHAPTER 10

Conclusions

Throughout the development of this final project I have learnt about the problems surrounding extremely dense networks and mobility. I have studied the solutions to tame these problems in order to continue the development of a testbed designed to implement these solutions by improving the mechanisms and adding new applications and APIs.

The experiments and test results obtained in the Experiments chapter (Chapter 6) prove that this project has successfully demonstrated the viability of the solution developed and implemented. The results could have been even better considering that, on one side, SDN being strongly dependent on software, the version of OpenFlow used is 1.0, and as explained in Chapeter 5.6 this brings many limitations. On the other side, SDN is also a strongly hardware-dependent technology, so if more powerful equipment had been used we believe we would have obtained better results.

I am absolutely delighted to have had the chance to be involved in such an exciting and ambitious project as this, exploring the future of wireless networks and telecommunications.

References

[1] E. Haleplidis, K. Pentikousis, S. Denazis, J. Hadi Salim, D. Meyer and O. Koufopavlou, *"Software-Defined Networking (SDN): Layers and Architecture Terminology"*. Internet Engineering Task Force (IETF). RFC 7426.

[2] C. Perkins, D. Johnson, and J. Arkko, *"Mobility Support in IPv6"*, Internet Engineering Task Force (IETF). RFC 6275.

[3] N. Feamster, J. Rexford, E. Zegura, "The Road to SDN: An Intellectual History of Programmable Networks," *ACM Queue*, Volume 11, Issue 12, 2013.

[4] "Software-Defined Networking: The New Norm for Networks," *Open Networking Foundation*, ONF White Paper, April 13, 2012

[5] F. Giust, L. Cominardi and C. Bernardos, "Distributed Mobility Management for future 5G networks: overview and analysis of existing approaches," *Institue IMDEA Networks and Universidad Carlos III de Madrid*, Spain

[6] S. Scott-Hayward, "Regulatory Implications of SDN & NFV: An ONF Perspective," *Open Networking Foundation*, 2016

[7] "OpenFlow Switch Specification," *Open Networking Foundation*, ONF White Paper, Version 1.3.0, October 14, 2013

[8] N McKeown, T Anderson, H Balakrishnan, G Parulkar, L Peterson, J Rexford, S Shenker and J Turner, *OpenFlow: Enabling Innovation in Campus Networks*, March 14, 2008

[9] H. Chan, D. Liu, P. Seite, H. Yokota and J. Korhonen, *"Requirements for Distributed Mobility Management"*. Internet Engineering Task Force (IETF). RFC 7333.

[10] G. Kirby, "Locating the User," *"Communications International"*, 1995

[11] F. Giust, A. de la Oliva, C. Bernardos, "Flat Access and Mobility Arquitecture: an IPv6 Distributed Client Mobility Management Solution," *Institue IMDEA Networks and Universidad Carlos III de Madrid*, Spain

[12] J. Zuñiga, C. Bernardos, T. Melia, A. de la Oliva, R. Costa and A Reznik, "Distributed Mobility Management: a Standards Landscape," *Communications Magazine, IEEE*, vol.51, March 2013

[13] S. Gundavelli, K. Leung, V. Devarapalli, K. Chowdhury and B. Patil, *"Proxy Mobile IPv6"*. Internet Engineering Task Force (IETF). RFC 5213.

[14] S. Sesia, I. Toufik, M. Baker, "LTE, The UMTS Long Term Evolution," *WILEY*, 2nd Edition, 2011

[15] "LTE-Advanced (3GPP Rel. 12) Technology Indtroduction," *ROHDE&SCHWARZ*, White Paper, Version 2e, August 4, 2015

[16] "LabVIEW Based Platform for Prototyping Dense LTE Networks in CROWD Project," *National Instruments*, White Papers, Jul 18, 2014

[17] "What Is PXI?," National Intruments, Official Web Page, Available at `http://www.ni.com/pxi/whatis/` accessed on Febuary 5th.

[18] "What Is ns-3?," nsnam, Official Web Page, Available at `https://www.nsnam.org/overview/what-is-ns-3/` accessed on Febuary 5th.

[19] M. Meeker, "INTERNET TRENDS 2015 – CODE CONFERENCE," *KPCB*, kpcb.com/InternetTrends, May 27, 2015

[20] M. Sanchez, A. Asadi, M. Dräxler, R. Gupta, V. Mancuso, A. Morelli, A. de la Oliva and V. Sciancalepore, "Tackling the increased density of 5G networks; the CROWS approach," *Vehicular Technology Conference, IEEE*, 11-14 May 2015

[21] H. Ali-Ahmad, C. Cicconetti, A. de la Oliva, V. Mancuso, M. Sama, P. Seite and S. Shanmugalingam, "An SDN-based Network Arquitecture for Extremely Dense Wireless Networks," *Future Networks and Services, IEEE*, 11-13 Nov. 2013

[22] "Study implications emerging technologies software defined networking and network function," *DIGITAL AGENDA FOR EUROPE*, European Commission, 2015

[23] "What is Ryu Controller?," Available at `https://www.sdxcentral.com/resources/sdn/sdn-controllers/open-source-sdn-controllers/what-is-ryu-controller/` accessed on Febuary 6th.

[24] "What is Open vSwitch?," Available at `http://openvswitch.org/` accessed on Febuary 6th.

[25] "VLC media player," Available at `http://www.videolan.org/vlc/index.html` accessed on Febuary 6th.

[26] "Pantou : OpenFlow 1.0 for OpenWRT," Available at `http://archive.openflow.org/wk/index.php/Pantou_:_OpenFlow_1.0_for_OpenWRT` accessed on Febuary 6th.

[27] "About Scapy," Available at `https://www.nsnam.org/overview/what-is-ns-3/` accessed on Febuary 6th.

[28] "InterDigital Europe and Partners to Lead Groundbreaking European Commission 5G Research Project," *InterDigital*, May 14, 2015

APPENDIX A

Summary

The main purpose of this final degree project was to continue, improve, expand and finish the development of a Software Defined Networking (SDN) based Distributed Mobility Management (DMM) solution under the Connectivity management for eneRgy Optimised Wireless Dense networks (CROWD) EU FP7 project[1]. This has been achieved by means of vastly increasing the functionality by implementing multi-node mobility support, mobile node whitelist, Infrastructure on Demand, Resource Detection, video streaming mobility, and error control; by integrating the solution with a LTE simulating network, a Video Server a Dynamic Backhaul Network emulated with a software program emulating SDN and graphical visualization of the network; and lastly by exhaustive testing and debugging of the code.

Wireless data communication is a constituent part of everyday life for hundreds of millions of people. From social networking to Internet-assisted navigation, from voice and infotainment to online gaming, mobile communication devices have become essential to fully live everyday interpersonal relations as well as participate in nationwide and world-wide events. The consequence is twofold. First, the number of wireless users is rapidly increasing, the offered load doubling every year, thus yielding a 1000x growth in the next ten years. Second, expecting high-quality services and high data rates is becoming normal rather than exceptional. For instance, considering a density population of 5000 people/km2, which is typical of large European cities like London, Madrid, or Paris, and accounting for 20% of the population being mobile data users, each demanding 1 Mbps, would lead to a demand of 1 Gbps/km2, which can be hardly provided by the current wireless infrastructures. The figure grows further if we consider that the per-user demand is expected to increase ten-fold in the next 5 years. Mobile data traffic increased by 81% in 2013 and by 69% in 2014 and the rate of increase is not slowing down (Figure A.1). 5G, the mmWave, is disruptive upcoming technology that employs frequencies in the order of 60GHz that will impose very high requirements, given the extremely high bandwidths it will provide in very small areas. This has a direct correlation with both the increased accessibility to the internet from mobile devices via WLANs and RANs, and the increase in popularity of applications designed for smartphones and tablets. On top of this, operators are migrating their networks to full IP based networks for both voice and data.

[1] `http://www.ict-crowd.eu/`

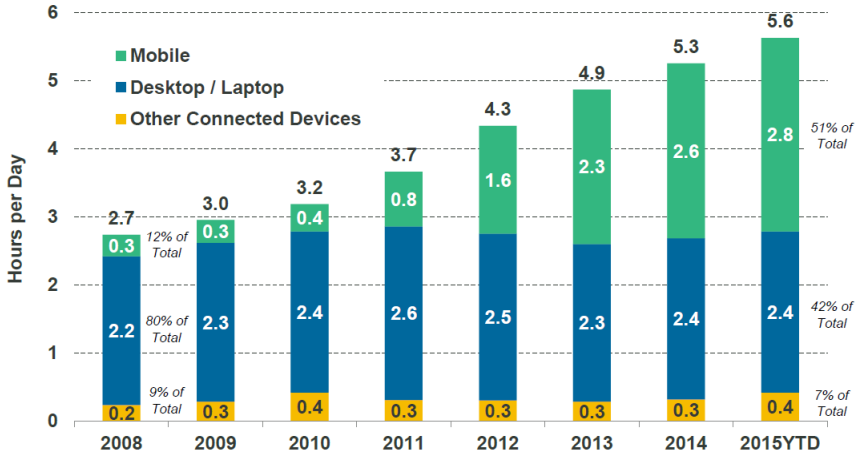


Figure A.1

The solution to cope with this growing traffic demand necessarily entails using more points of access, by increasing their density (dense network deployments) and/or by using different wireless technologies (heterogeneous deployments). Following this trend, network operators have already started to push for denser deployments, building micro-, pico- and femto-cells, and installing Wi-Fi hotspots in public areas to inject capacity where the data traffic demand is particularly high. These efforts notwithstanding, we argue that increasing the number of points of access alone would not remove capacity and performance bottlenecks. In fact, dense deployments are not necessarily synonymous with higher capacity. The case of smart meters is a key example. It has been recently noticed that the diffusion of meters for gas and electricity, endowed with wireless transmitters using the 2.4 GHz ISM band, is generating erratic behaviour in Wi-Fi home devices in USA.Furthermore, having a large number of deployed access points also influences the energy cost, especially for the network operator. In particular, today's access points and base stations running at zero-load consume almost as much energy as when running at full capacity. As a result, wireless dense networking can potentially lead to wireless chaos and huge energy waste. The ever-growing demand for wireless connectivity requires more and more capacity, in terms of both data volume and number of connected devices. Wireless, dense, and heterogeneous deployments are today's only viable solution to provide such capacity. Since the design of currently available technologies did not account for density- and heterogeneity-related issues, the latter pose new, unexplored, and promising research questions. Specifically, we claim that using the existing wireless technologies in dense scenarios will require new mechanisms for coordination and cooperation of wireless devices; otherwise we will soon witness the implosion of wireless network performance and the explosion of network operation costs.ility Management (DMM) as a networking solution to the mobility and network management problems in extremely dense wireless networks.

SDN is an approach to networking that involves the separation of the control plane from the data forwarding plane, resulting in the capacity to manage network behaviour dynamically. This is achieved through the abstraction of the forwarding plane using standardized interfaces. The offers the capability of configuring the network backhaul

from a centrilized entity without having to configure each element independently. In this project we use OpenFlow as the protocol for providing the communication between the CLC and the configurable backhaul running the OpenFlow Switch module. The OpenFlow Switch consists of a Flow Table, an Action associated with each Flow Table Entry and an OpenFlow Secure Channel. The Flow Table performs packet lookups and forwarding, and the OpenFlow Secure Channel connects the OpenFlow switch with the SDN Controller. The switch communicates with the controller and the controller manages the switch via the OpenFlow protocol

DMM is a an approach to IP mobility that suggests a distributed network structure compared to a centralized structure Distributed mobility management (DMM) is an approach to evolve from centralized mobility protocols to more distributed protocols, based on already existing centralized mobility stadardized protocols such as Mobile IPv6, Hierarchical Mobile IPv6 and Proxy Mobile IPv6.

Problems that are solved with a DMM implementation are:

- **Sub-optimal routes:** Since mobility anchors are fixed at the home address, this is where all traffic will arrive and must be forwarded from to the current mobile node's address. This results in longer end-to-end paths meaning higer delays. With a distributes mobility architecture, as the anchors are located at the very edge of the network, close to the user terminal, data paths tend to be shorter.
- **Lack of scalability:** A centralized entity in charge of maintaining mobility context information for each mobile node needs to have enough processing and routing capabilities to deal with all the mobile users' traffic simultaneously. DMM suggests distributing the load among several network entities.
- **Single point of failure and attack:** A centralized entity in charge of maintaining mobility context information for each mobile node means there is a unique point of failure and attack.
- **Unnecessary mobility support:** IP mobility support is usually provided to all mobile nodes, even for users that will not move from the first point of attachment. This also applies to sessions that deal with mobility at an application level or applications that don't need a stable IP address during a handover to maintain session continuity.

The solution proposed to deal with the problems discussed in this section, is a Connectivity management for eneRgy Optimised Wireless Dense networks (CROWD) initiative, that uses a combination of Software Defined Networking and Distributed Mobility Management techniques. The CROWD Architecture (Figure A.2) is structured into two logical tiers, each managed by a CROWD Controller: districts with a limited, but fine grain scope for short time scales and operated by a CROWD Local Controller (CLC), and Regions with a broader but more coarse grain scope for long time scales and operated by a CROWD Regional Controller (CRC).

The approach to solve the mobility issue is by following the Distributed Mobility Management approaches which are, on one side, transforming Mobile IPv6 into a

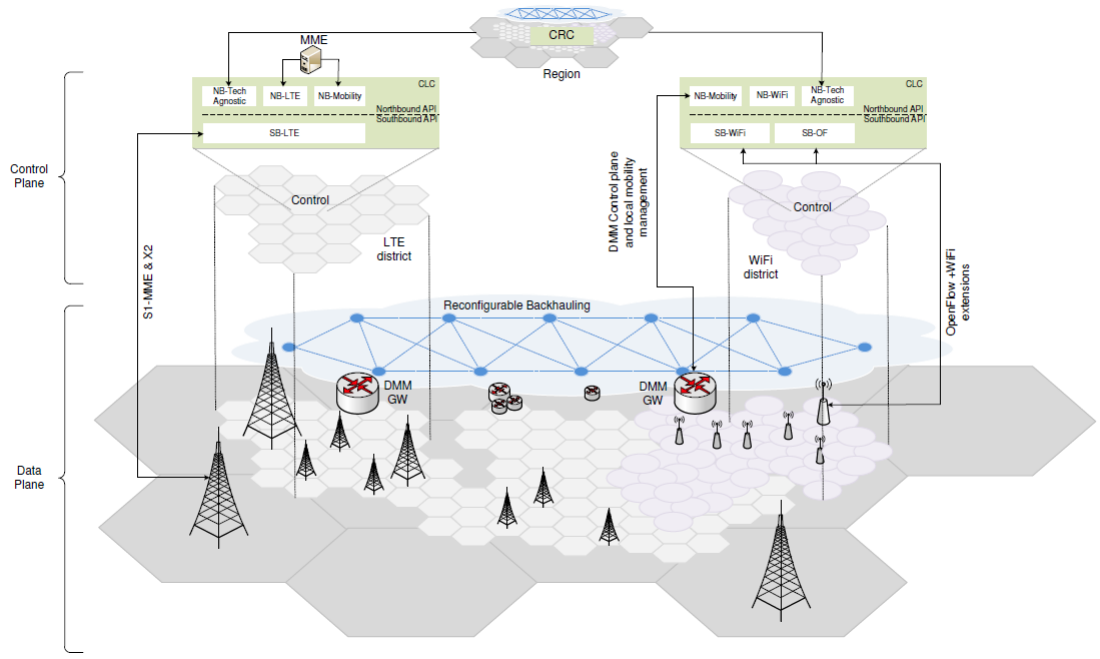


Figure A.2

distribution-oriented protocol, and, on the other, doing the same for Proxy IPv6. These approaches can be identified as client-based or network-based solutions respectively:

- **Client-based solution:** Based on the Mobile IPv6 protocol, this solution involves deploying multiple home agents at the edge of the network in order to distribute the mobility anchors. This is achieved through the idea that a mobile node can have assigned more than one IP address, meaning that the mobile node can have assigned a new IP address every time it visits an access network.
- **Network-based solution:** Based on the Proxy IPv6 protocol, this solution suggests moving the mobility anchors to the edge of the network, and give them the capability for managing both the control and data plane separately. The ability of managing both planes separately offers the possibilty for optimal routing of data traffic.

Our implementation consists of two separate but not independent networks we call districts. Each district is controlled by an SDN local Controller and an OpenFlow controlled configurable backhaul accessed by 802.11 capable Points of Attachment (PoA). The basic functionality of the system is structured as follows:

- **Attachment**: The mobile node associates to an Access Point (AP) belonging to a CROWD District for the first time.
- **Video streaming**: The mobile node sends a Start Video request to the Video Server.
- **Intradistrict mobility**: The mobile node performs a handover to an AP belonging to the same district

- **Interdistrict mobility**: The mobile node performs a handover to an AP of a new district
- **Mobility to a non-CROWD network**: The mobile node performs a handover to a PoA belonging to a non-CROWD network.

The improvements made to the project from the moment it was handed to us to continue the development are:

- **Multi-node support**: provision of mobility support and connectivity to one or more mobile node
- **Mobile node whitelist**: a list of MN's allowed to attach
- **Resource Detection**: the capability of detecting the nodes active in the district
- **Video streaming capability**: provision of mobility support when streaming video
- **Infrastructure on Demand**: resource management, activating or deactivating APs following certain criteria
- **Error control**: sometimes during a handover, certain packets misfire and provide unvalid information.

The technologies that have been integrated with the implementation of the solution proposed since the project was handed to us are:

- **LTE simulation:** A major part of this final project was integrating LTE capabilities with the system.e have three laptops, two representing the UE and one the enB. From the two laptops representing the UE, one runs the GUI used for simulating handovers and the other runs the UE side of the ns-3 program. The eNB runs the eNB side of the ns-3 program, the S-GW and the PDN-GW. If using the ns-3 program for LTE simulation, the LTE connection and the data transmission is completely simulated. If we also use the NI PXI, then LTE connection and data transmission is performed over a real channel using real transceivers.
- **Video Server:** The Video Server, implemented in a Raspberry Pi, runs a python UDP server that accepts request messages for video streaming. When a MN wants to send a Video Start request message, a first NOP (No Operation) message must be sent because it will be consumed by the CLC for processing and will never reach the Video Server if there are no rules installed for the MN in the OF Switches. When the Video Start messages reaches the Video Server, it uses a VLC command to start streaming to the address that has sent the request using Real-time Transport Protocol (RTP), a network protocol for delivering audio and video over IP networks. As soon as the request is sent, VLC is started on the MN awaiting the UDP video stream.
- **MaxiNet:** a Dynamic Backhaul Network emulation extension of the Mininet environment, created to span the emulation across several physical machines to allow the emulation of very large software-defined networks. As a part of this final

project, we successfully integrated our system with a MaxiNet implementation acting as a blackbox, to validate the reachability of the Video Server through a unknown SDN network.

- **Visualization:** A fundamental part of this project is the integration of real-time visualization of the network and its state. This has been achieved by developing a Web Site with its respective Web Server, that get the network information from the CLCs using an API. This has the capacity of illustrating:
  - **Resource detection**
  - **MN attachment**
  - **Datapath installation**
  - **Mobility**
  - **Tunnel creation/deletion**
  - **Infrastructure On Demand**

Taking into consideration that this project is a *proof of concept*, in order to prove the correct functionality of the system and measure the performance, fully comprehensive experiments and tests have been carried out thoughout the project. The experiments were implemented using bandwidth performance tools such as iperf and using Wireshark as a network protocol analyzer. One of the experiments carried out measures the time an MN takes to join a CROWD district and connects with its assigned DMMGW, represented by a cumulative distribution function (Figure A.3 and A.4) and by a bar chart (Figure A.5). The same has been done for intradistrict mobillity, interdistrict mobility, and mobility to a non-CROWD network. Infrastructure on Demand was tested by using iperf and Wireshark, measuring the thoughput going through the AP and two MNs connected to is, and how it changes when a MN is forces to perform a handover to a newly switched on AP.

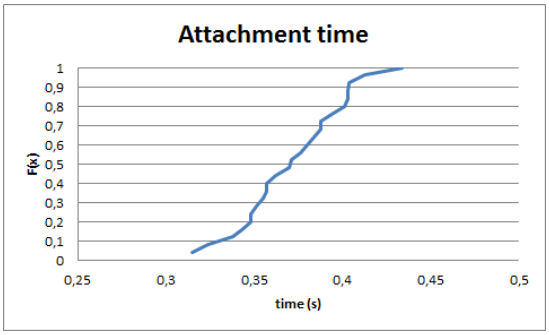


Figure A.3

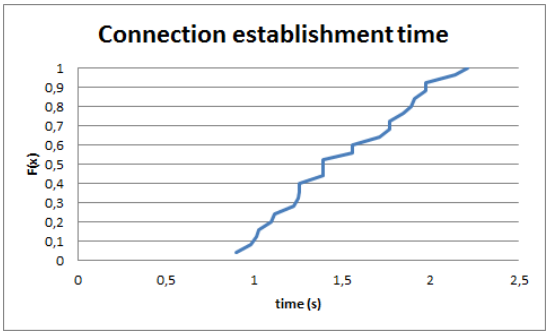


Figure A.4

Regarding the regulatory framework, all the OpenFlow nodes used in the project are IEEE 802.11 compliant and the code follows the OpenFlow 1.0 spcification. The European Commission has commissioned a study on the "Implications of the emerging technologies Software-Defined Networking and Network Functions Virtualization on the future Telecommunications landscape". One of the objectives of the study is to explore the unknown area of technological socio-economic-regulatory impact of Software-Defined Networking. The study aims to:

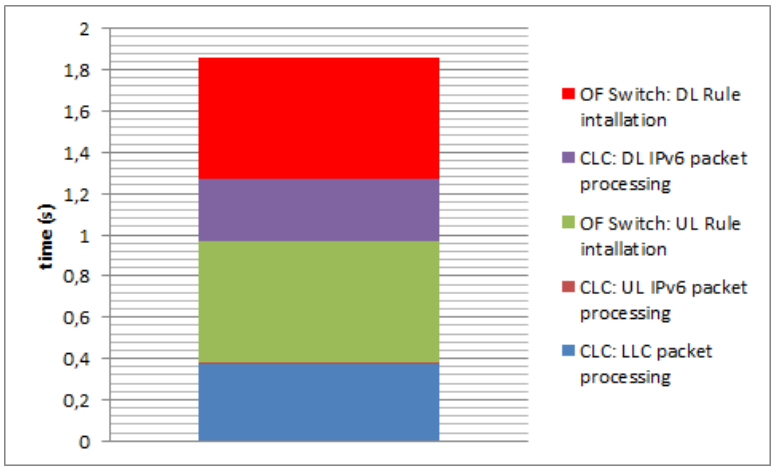


Figure A.5

- Identify the most likely deployment scenarios including associated timelines and possible migration paths from existing networking technologies
- Identify current and new usage scenarios of SDN-based networks including trends and possible new services
- Assess the impact of SDN on existing business models in the telecommunications sector and identify their innovation potential in terms of possible new business models with current and new stakeholders
- Assess the impact of SDN on existing business models in the telecommunications sector and identify their innovation potential in terms of possible new business models with current and new stakeholders
- Assess the general market/industrial potential for SDN.
- Position SDN within the current and future telecommunications regulatory framework and in relation to growth opportunities.

InterDigital Europe, the company's London-based 5G-focused research unit, has been chosen by the European Commission to conduct a 5G socioeconomic research project [28]. The research will directly inform the development and rollout of 5G in the European Union, and empower the region to stake an early claim as a global 5G leader. The study will produce an unprecedented framework, based on key socioeconomic data, for the European Commission's introduction of 5G infrastructure in Europe. The project will engage key stakeholders across the wireless industry, as well as vertical markets and policy makers, to get a clear picture of 5G's economic impact and the possible innovation it will spur in areas such as health, transport, social services and commercial opportunities. Executed over the next year, the study will be completed by InterDigital in partnership with additional 5G experts including Real Wireless, Rethink Wireless, Tech4i2 and Trinity College, Dublin. The study will:

- Map out possible 5G scenarios in the region's economy and society to determine technological requirements and parameters for 5G implementation.

- Quantify rural, suburban and urban usage patterns and their impact on technology requirements
- Identify expected use cases for 5G technology, and their impact on technical parameters
- Analyze spectrum demand and technical spectrum issues, including quantitative needs and boundaries, and consider new band requirements and constraints